(19)日本国特許庁（ＪＰ）　(12)公表特許公報（Ａ）　(11)特許出願公表番号
特表2002－518899
（P2002－518899A）
(43)公表日　平成14年６月25日(2002.6.25)

(51)Int.Cl.[7]	識別記号	ＦＩ		テーマコード(参考)
Ｈ０４Ｎ　5/91		Ｇ１１Ｂ　20/10		Ｈ　５Ｃ０５３
Ｇ１１Ｂ　20/10			20/12	５Ｃ０５９
20/12		Ｈ０４Ｎ　5/91		Ｐ　５Ｄ０４４
Ｈ０４Ｎ　5/92			5/92	Ｈ
7/24			7/13	Ｚ

審査請求　未請求　予備審査請求　有　（全110頁）

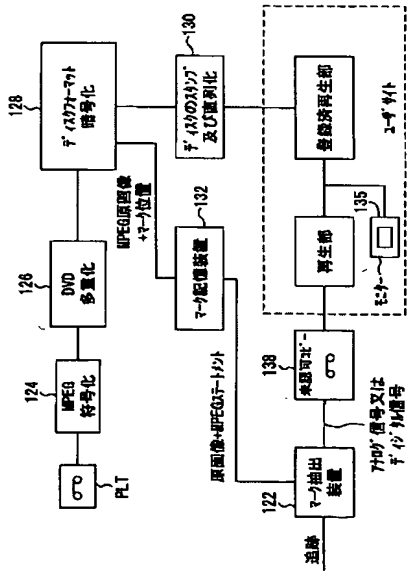
(21)出願番号　特願2000－554137(P2000－554137)
(86)(22)出願日　平成11年６月７日(1999.6.7)
(85)翻訳文提出日　平成12年12月８日(2000.12.8)
(86)国際出願番号　ＰＣＴ／ＵＳ９９／１１７９７
(87)国際公開番号　ＷＯ９９／６５２４１
(87)国際公開日　平成11年12月16日(1999.12.16)
(31)優先権主張番号　０９／０９２，８９８
(32)優先日　平成10年６月８日(1998.6.8)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　ディジタルビデオエクスプレス　エル．ピー．
アメリカ合衆国　バージニア州　20170、ハーンドン、ハーンドンパークウェイ　580
(72)発明者　ロバート　ダブリュー．　シューマン
アメリカ合衆国　バージニア州　22124　オークトン、クエイロード　11723
(72)発明者　シューレオン　イウ
アメリカ合衆国　バージニア州　22043　フォールズチャーチ、202、ピンミットランレーン　2252
(74)代理人　弁理士　前田　弘　（外７名）

最終頁に続く

(54)【発明の名称】　動画像等の前もって記録された所有権主張材料の非認可コピーの出所を追跡する装置及び方法

(57)【要約】
ビデオＤＶＤ等の情報含有媒体の非認可コピーの出所を追跡可能にするため、媒体に記録された圧縮ディジタル信号ストリームを、フレーム毎に位置が変化しコピーが製造された場所と時間を示す情報で符合化された画素ブロックを有する「走行マーク」を含むように変更する。検査対象媒体を元のビデオ信号と走行マークの位置とを含む基準媒体と同時に再生する。コピーメッセージの出所を有利に暗号化及びスクランブル化してコピー者による検出と変更を防止する。メッセージを、例えば、スペクトラム拡散キャリアに拡散させて、多数の出所の中からの識別性を向上させるとともに、ノイズが存在する中での復調を可能にする。



【特許請求の範囲】

【請求項１】　情報信号源により生成された情報含有圧縮ディジタル信号ストリームを処理して、オリジナル記録媒体に記録し、情報含有信号ストリームの情報内容に印を付けてオリジナル記録媒体からの情報含有圧縮ディジタル信号ストリームの非認可コピーの出所を追跡可能にする方法であって、

上記オリジナル記録媒体の再生時に元の圧縮ディジタル信号ストリームを受け取る工程と、

上記オリジナル記録媒体から上記元の圧縮ディジタル信号ストリームの再生の出所に対応するコピー源メッセージを得る工程と、

上記コピー源メッセージを含むよう上記元の圧縮ディジタル信号ストリームを修正することにより修正信号ストリームを生成する工程とを備えた方法。

【請求項２】　上記修正信号ストリームが第２の記録媒体で再生成される請求項１記載の方法であって、

上記第２の記録媒体の再生時に上記修正信号ストリームを再生成する工程と、

上記元の情報含有圧縮ディジタル信号ストリームを上記修正信号ストリームと比較して上記コピー源メッセージを抽出する工程とを備えた方法。

【請求項３】　上記情報含有圧縮ディジタル信号ストリームはビデオ信号であり、上記圧縮ディジタル信号ストリーム修正工程は上記圧縮ディジタル信号ストリームにより生成された画像の特徴を終始するものである請求項１記載の方法。

【請求項４】　上記圧縮ディジタル信号はＭＰＥＧ圧縮標準で前もって符号化される請求項３記載の方法。

【請求項５】　修正用にビデオフレームから少なくとも一つの領域を選択し、その選択されたビデオフレーム領域を修正する工程を備えている請求項４記載の方法。

【請求項６】　選択されたビデオフレーム領域のアドレスを格納する工程を備えている請求項５記載の方法。

【請求項７】　上記選択はビデオフレーム内の画像の画像属性に基づいて実行される請求項６記載の方法。

【請求項８】　上記画像属性は画像境界線である請求項７記載の方法。

【請求項９】　上記選択されたフレーム領域を上記コピー源メッセージに対応する修正フレーム領域に置換する工程を備えている請求項６記載の方法。

【請求項１０】　上記選択されたフレーム領域を修正フレーム領域への置換の前に調整する工程を備えている請求項９記載の方法。

【請求項１１】　上記コピー源メッセージに応じて連続するフレームから修正用に様々な領域を選択する工程を備えている請求項５記載の方法。

【請求項１２】　上記フレーム領域は画素ブロックまたはマクロブロックを備えている請求項１０記載の方法。

【請求項１３】　上記圧縮ディジタル信号ストリームの修正前に上記コピー源メッセージを暗号化する工程を備えている請求項１記載の方法。

【請求項１４】　上記圧縮ディジタル信号ストリームの修正前に上記コピー源メッセージをスクランブル化する工程を備えている請求項１記載の方法。

【請求項１５】　上記圧縮ディジタル信号ストリームの修正前に上記コピー源メッセージを符号化してスペクトラム拡散キャリアを生成する工程を備えている請求項１記載の方法。

【請求項１６】　予め記録されたオリジナル媒体から再生されたオリジナルＭＰＥＧ符号化ビデオ信号ストリーム及び／または音声信号ストリームの形で圧縮ディジタル情報を処理して上記媒体からの信号ストリームの非認可コピーを追跡可能にする方法であって、

ＭＰＥＧ標準形式の情報とその後の再生を記述する情報に関するデータビット用のビデオフレーム内の前選択された画素ブロック位置を格納するオリジナル媒体を生成する工程と、

上記オリジナル媒体のその後の再生時にＭＰＥＧデータストリームを受け取る工程と、

上記ＭＰＥＧ信号ストリームを修正することにより修正信号ストリームを生成して上記その後の再生情報を上記前選択された画素ブロック位置に対応する画素ブロックに埋め込む工程とを備えている方法。

【請求項１７】　上記元の信号ストリームの内容をオリジナル媒体の非認可

コピーであると疑わしい記録媒体から再生された修正信号ストリームの内容と比較して再生情報を抽出する工程を備えている請求項１６記載の方法。

【請求項１８】　上記再生情報は記録媒体識別、再生成装置識別及びデータストリーム再生成時のうちの少なくとも一つを含んでいる請求項１６記載の方法。

【請求項１９】　修正用のメッセージ領域としてフレームの少なくとも一つの領域を選択し、該選択された領域を修正する工程を備えている請求項１６記載の方法。

【請求項２０】　上記選択は画像属性に基づいて実行される請求項１９記載の方法。

【請求項２１】　上記画像属性は画像境界線である請求項２０記載の方法。

【請求項２２】　上記選択されたフレーム領域を上記再生情報の要素を指定する修正フレーム領域に置換する工程を備えている請求項１９記載の方法。

【請求項２３】　上記選択されたフレームを修正フレーム領域への置換の前に調整する工程を備えている請求項２２記載の方法。

【請求項２４】　連続するフレームから修正用に様々な領域を選択する工程を備えている請求項１９記載の方法。

【請求項２５】　上記領域は画素ブロックまたはマクロブロックを備えている請求項１９記載の方法。

【請求項２６】　上記再生情報をＭＰＥＧ信号ストリームに埋め込む前に上記画素ブロックまたはマクロブロックをスクランブル化する工程を備えている請求項２５記載の方法。

【請求項２７】　上記ＭＰＥＧ信号ストリームを修正する前に上記再生情報を暗号化する工程を備えている請求項１６記載の方法。

【請求項２８】　上記ＭＰＥＧ信号ストリームを修正する前に上記再生情報を符号化してスペクトラム拡散キャリアを生成する工程を備えている請求項１６記載の方法。

【請求項２９】　上記ＭＰＥＧデータストリームのＢフレームのみが置換用フレーム領域として選択される請求項２２記載の方法。

【請求項３０】　上記ＭＰＥＧ信号ストリームを修正する前に上記再生情報の誤りを訂正する工程を備えている請求項１６記載の方法。

【請求項３１】　オリジナル記録媒体から再生されたＭＰＥＧ符号化ビデオ信号ストリーム及び／または音声信号ストリームの形で圧縮ディジタル情報を処理して上記オリジナル媒体からの信号ストリームの非認可コピーを追跡可能にする装置であって、

上記圧縮信号ストリームを含むオリジナル媒体の再生の出所に関するコピー源メッセージを得る検出器と、

上記オリジナル記録媒体からの再生時に上記ＭＰＥＧ信号ストリームを受け取るレシーバと、

上記コピー源メッセージを上記ＭＰＥＧ信号ストリームに埋め込むコンバイナとを備えている装置。

【請求項３２】　上記ＭＰＥＧ符号化信号ストリームの選択されたビデオフレーム内の画素ブロックの位置を格納するメモリを備えている請求項３１記載の装置。

【請求項３３】　上記検出器により受け取られたコピー源メッセージをスクランブル化するスクランブラを備えている請求項３１記載の装置。

【請求項３４】　上記検出器により受け取られたコピー源メッセージを暗号化する暗号化部を備えている請求項３１記載の装置。

【請求項３５】　上記コピー源メッセージを上記ＭＰＥＧ信号ストリームに埋め込む前に上記コピー源メッセージを符号化してスペクトラム拡散キャリアを生成するエンコーダを備えている請求項３１記載の装置。

【請求項３６】　元のデータストリームからコピーされたデータストリームを圧縮ディジタル信号の形で処理してコピーの出所に対する追跡を可能にする方法であって、

上記元のデータストリームを受け取る工程と、

所定サイズのメッセージホールを上記元のデータストリーム内の予め定義された不規則に発生する位置に配置する工程と、

上記メッセージホールに埋め込まれたコピー源メッセージを上記データストリ

ームから抽出する工程とを備えている方法。

【請求項３７】　記憶媒体に記録されるデータ含有ディジタル信号を処理して上記データの非認可コピーをコピー者まで追跡できるようにする方法であって、

上記ディジタル信号ストリームが格納された媒体から上記ストリームを再生することによりディジタル信号ストリーム含有データを受け取る工程と、

上記データの再生の出所に関するコピー源メッセージを得る工程と、

上記コピー源メッセージを含んだスペクトラム拡散キャリアを生成する工程と、

上記符号化されたコピー源メッセージを上記ディジタル信号ストリームと結合する工程とを備えている方法。

【請求項３８】　上記ディジタル信号ストリームが圧縮される請求項３７記載の方法。

【請求項３９】　上記ディジタル信号ストリームがＭＰＥＧ圧縮標準形式で符号化される請求項３８記載の方法。

【発明の詳細な説明】

【０００１】

技術分野

本発明は一般に、情報―関連媒体（例えばＤＶＤ）等のソースから得たデータのためのコピー保護に関し、さらに詳しくは、無許可でコピーする者が確認できるようにするためのデータ修正に関する。

【０００２】

背景技術

映画、ソフトウェア等の記録されたプログラム資料の録画、及び再生のための消費者用装置は広く入手可能である。そのような資料に関する媒体は、磁気テープ、ＶＣＲを使用した録画／再生を含み、また最近では、ＣＤ、ＣＤ－ＲＯＭやＤＶＤといった光学媒体もある。ＤＶＤは、新しい光学ディスク技術であり、各層で一般に１３３分の映画のための十分な情報を保持できる。ＤＶＤは、ビデオ資料を効率よく記憶し、目に見えるような劣化なく再生できる、いわゆるＭＰＥＧ－２データ圧縮基準をしばしば使用している。ソフトウェアで実施しているビデオや多種の情報のための他の基準も利用できる。

【０００３】

娯楽映画等のプログラム資料の製造コストはかなりのものであるが、コピーする者のコストは比較的安い。したがって、映画の著作権侵害が映画産業の収入源の重大な損失となっている。実際、コピーする者、主に不法ビデオカセットによって年間２０億ドル以上を損失している。ＤＶＤプレーヤからのオーディオ、ビデオプログラムには高度な忠実性があり、ＶＨＳからのものよりもかなり多くのコンテンツを保持できるので、ＤＶＤはこのような状況をよりひどくしている。

【０００４】

著作権侵害を妨害する２つの試みが映画の合法製作者と商業価値のある他の業種によって開発されており、そのソフトウェアとは：（１）再生時にある方法で資料をひずませる信号を保護された媒体に注入するもの、及び（２）可視、又は不可視の静止符号、又は元の資料のソースを確認するマークを注入して隙間を入れるものである。

【０００５】

隙間は、一般に保護された資料を書く時点で媒体に入れるものである。例えば、情報―関連媒体が製作される郵便局や他の施設である。しかし、一度記録すると、透かしは固定されてしまう。隙間は資料を書くソースを確認できるが、無許可でコピーする者の確認に関する情報はない。コピーする者の確認は再生、コピーする時点で収集された情報から得られなくてはならない。これは、コピーするものが使用する装置と、コピーする時間の確認としての情報を含む。しかし、再生ユニットでこの情報を提供する電気回路は、ユニットのコストを吊り上げてしまい、競合産業において深刻な欠点となる。本発明の目的は、使用する時点で、保護資料のコピーするものによって変化し、保護資料にとって唯一の追跡信号を生成することである。本発明の他の目的は、再生ユニットのコストをさほど上げることなくそうすることである。本発明のさらなる目的は、媒体のドメインによって、デジタル又はアナログスペースでこの機能を果たすことである。

【０００６】

予測可能性が信号をソースの偽造指定をするようフィルターで取り除く、又は変えることができるようにしてしまうことになるため、保護信号はさらに予測できなければならない。さらに、信号ストリームに注入されるコピーメッセージのソースを重い送信ノイズや、意図しない又は意図的なひずみのある環境で復元しなければならないため、保護システムは頑丈でなければならない。さらに、送信メッセージは、検出、改変、除去しにくくなくてはならない。メッセージのコンテンツは、送信ビット認識ソースがあっても、保護され、復号しにくくなくてはならない。最後に、出力ストリームは、適切なドメインで正当に復号でるよう合法でなければならない。本発明のさらなる目的は、保護された媒体の信号ストリームに注入した、コピーする者が検出又は変更しにくい使用信号のポイントを生成することにある。さらに使用信号のポイントは、ビデオに加えてオーディオやソフトウェアを含む保護するあらゆる種類のコンテンツに注入できるようにする。

【０００７】

発明の開示

本発明によれば、情報―関連圧縮デジタル信号ストリームを処理する方法は、“コピーのソースメッセージ”をビデオフレームのピクセルブロック又はマクロブロックに配分された“ランニングマーク”の形式で加えることによりストリームれを改変し、無許可コピーのソースを追跡できるようにするものである。正常使用における改変信号ストリームは、使用者が見るためのテレビ表示装置又はモニタに送出される。ランニングマークは薄く、普通は目に見えない。しかし、信号ストリームを無許可で録画するために使用する場合は、記録にはコピーのソースメッセージと共に元の圧縮デジタルストリームが与えられる。したがって、元のデジタル信号ストリームと改変信号ストリームを比較すれば、メッセージが抽出でき、コピーした者を追跡できる。

【０００８】

好ましい実施例では、例えば、圧縮デジタル信号ストリームは、ＭＰＥＧ基準で符号化されたオーディオ／ビデオデータストリームである。ピクセルブロック、又はマクロブロックは、コピーメッセージのソースを運ぶための“メッセージホール”という候補領域としてフレームベースで選択される。メッセージホールは、ビット調整で変更可能な画像フレームのブロック又はマクロブロックに関するＭＰＥＧビットストリームの選択されたセグメントに対応している。

【０００９】

コピーのソースメッセージは、予め記録された資料（映画等）の再生時に置換ビットとして信号ストリームに与えられる。メッセージホールとなるよう選択されたマクロブロックは様々な方法でビット調整される。好ましい実施例では、例えば、調整は、好ましくは（ＤＣＴ）ディスクリートコサイントランスフォームドメイン（あるいはビデオドメイン）で規定量のノイズを選択されたマクロブロックに加算、又は選択されたマクロブロックから減算して行う。これにはマクロブロック内に歪を広げ、ピクセル歪を見えにくくするという利点がある。

【００１０】

メッセージホールは、フレーム内にランダムに位置しているが、使用者には見えない又は殆ど見えない改質を調節する、画像境界等の画像の織目部分に存在するよう選択されることが好ましい。候補メッセージホールの位置は、オペレータ

の入力又は機械操作で、例えばフレームごと（又は一度にマルチプルフレーム）に選択してもよい。選択された位置は再分布されて、フレーム内でより均一なメッセージホールの分布となる。再分布はメッセージホールの数をフレームごとのメッセージホールの目標数に近づけるよう保持することに基づき、１フレーム又は少数の隣接するフレーム内で密集しないようにする。

【００１１】

行内の選択されたマクロブロックに含まれるビット数を相対的に一定に保つために、ランニングマークを付加した後に、ビット数をＭＰＥＧ基準に一致させ、バッファの底流又は流出を防ぐためにＭＰＥＧ再符号化をして補間してもよい。これは、例えば、マクロブロックビットの元の数とランニングマークで符号化した後のビット数が略同じになるまで、マクロブロックピクセルの量子化スケール要素を変更すればできる。

【００１２】

メッセージホール候補の選択に続いて、再符号化処理中に、交互メセージホールが規定の基準を満たしているか、コピーメッセージビットのソースを挿入するために使用すべきかを決定するために、候補メセージホールをテストし、両方を受け入れ、放棄し、再分布する。不適切な、又はおそらく不適切であろうメッセージホールとしては、例えば：ＤＶＤ区分境界にある行のメッセージホール、ビット補間で再符号化できない行のメッセージホール、確実に復号できないメッセージビットを含むメッセージホール、及び再符号化に続いてスキップされるメッセージホールがある。

【００１３】

コピーのソースメッセージは、以下の１つ以上に対応している：無許可でコピーをする元の許可再生ユニットの連続番号、コピーする媒体の連続番号、及びコピーの時間。この情報は、再生ユニットと記録媒体を提供する機関によって保持される記録を使用して、無許可でコピーする者を確認する助けとなる。

【００１４】

無許可でコピーする者がその情報をフィルターで除去、又は改変する目的でコピーのソースメッセージを復号するのを防ぐ助けをするために、メッセージは記

号化し、混乱させるのが好ましい。コピーする者の追跡を困難にする多数の再生ユニットで生成したコピーのソースメッセージビットを相関関係検出できるコード分割マルチプルアクセス符号化（ＣＤＭＡ）を使用してメッセージを符号化すると便利である。

【００１５】

メッセージホールのランニングマークコンテンツを抽出し、復号化するためには、各メッセージブロックの位置をメモリに記憶させる。関連媒体は、メッセージホールの位置と共に元のプログラム資料に付けて生成する。コピーのソースを確認するためには、テストで媒体と同期して関連媒体を再生する。対応する位置でテストで媒体から読まれ、関連媒体からも読まれたピクセルブロックをお互いに比較し、コピーのソースメッセージビットで改変された元の画像かどうか判断する。例えば、逆でもよいが、メッセージホールでの画像改変は、“１”、改変がないものは“０”と指定する。２進法フォーマット以外を含む他の情報フォーマットは選択的に実施してもよい。

【００１６】

本発明の特異な目的、利点、新規の特徴は、以下に続く説明の部分で述べ、ある部分では、当業者が以下を検査すれば明らかとなり、本発明の実施によってわかるであろう。本発明の目的及び利点は、添付のクレームで特に指摘した手段、及び組合せによって達成できることがわかる。

【００１７】

図面の簡単な説明

図面について説明するが、明細書中、同様の符号を付した構成要素は同様の構成要素を表し：

図１は、本発明によるＭＰＥＧランニングマーク位置の作成を示す高レベル機能ブロック図である。

【００１８】

図２は、コピーのソースメッセージを行う３つのメッセージホールのある画像を示す。

【００１９】

図３は、ビデオストリームで実施したメッセージホールを示す線図である。

【００２０】

図４（ａ）、及び４（ｂ）は、本発明によるランニングマークシステム符号器及び復号器を示すブロック線図である。

【００２１】

図５は、データ記号基準を使用したメッセージビットの記号化を示す線図である。

【００２２】

図６（ａ）、及び６（ｂ）は、鎖状誤差訂正符号化及び複合化の線図である。

【００２３】

図７は、本発明の実施例で実行する包旋状誤差訂正コーダを示す。

【００２４】

図８は、本発明の実施例で実行するＣＤＭＡ符号化処理を示す線図である。

【００２５】

図９（ａ）－（ｅ）は、ビデオストリームで実施したコピーのソースメッセージのセキュリティを向上させるためのマルチプル可変長配行を示す。

【００２６】

図１０（ａ）－（ｄ）は、本発明によるビット符号化の変形例を示す

図１１は、本発明に一形態でのＤＣＴキャリア係数へのノイズ組成を加えることによる元の信号波形の広がりを示す。

【００２７】

図１２は、本発明におけるＲＭカメラ生成のためのＤＣＴの試みを示す擬似コードである。

【００２８】

図１３は、ＤＣＴの試みを使用した輝度ブロックのためのピクセルドメインにおけるノイズパターンを示す。

【００２９】

図１４（ａ）－（ｄ）は、ランニングマークデータストリームのレイアウトと構成の線図である。

【００３０】
図１５は、本発明によるＤＶＤ信号ストリームで行われるビット抽出を示す線図である。
【００３１】
図１６は、本発明によるランニングマーク生成の動作を示すアルゴリズムである。
【００３２】
図１７、及び１８は、本発明の一形態によるＤｉｖｘマーク挿入前、挿入後のビデオグループのレイアウトを示す。
【００３３】
図１９は、ｇＭＨ行を定義するアルゴリズムである。
【００３４】
図２０は、本発明に一形態によるマクロブロックの分類のための擬似コードである。
【００３５】
図２１は、分類機能によって要求される初期化（）のための擬似コードである。
【００３６】
図２２は、マクロブロック分類の他の実施例のための擬似コードである。
【００３７】
図２３は、処理する典型的なフレーム区分の線図である。
【００３８】
図２４は、再分配手順の擬似コードである。
【００３９】
図２５は、再分配機能によって要求される初期化（）のための擬似コードである。
【００４０】
図２６は、メッセージホールの予備ビットの補間するための流れ行におけるマクロブロックのためのＭｑｕａｎｔの増加数値を示す。

【００４１】
図２７は、本発明によるＭＰＥＧ再符号化の第一形態を示すアルゴリズムである。
【００４２】
図２８は、第２形態を示すアルゴリズムである。
【００４３】
図２９は、ｋ回（ｋ＝（ｄ＋２））繰り返すためのｊの位置決めのアルゴリズムである。
【００４４】
図３０は、ＭＰＥＧ再符号化の第３形態を示すアルゴリズムである。
【００４５】
図３１は、本発明によって行われる機能をまとめた高レベル線図である。
【００４６】
図３２は、ＧｅｎＲＭプログラムを示すアルゴリズムである。
【００４７】
図３３は、見る角度を変えたビデオ表示を示す。
【００４８】
表の簡単な説明
表１は、ｎ＝８の場合のアダマールーウォルシュ波形を示す。
【００４９】
表２は、列と行の配列後のアダマールーウォルシュ波形を示す。
【００５０】
表３は、別に規定したアプリケーションのプロバイダ決定ストリームのためのＤｉｖｘＳｔｒｅａｍＩｄを示す。
【００５１】
表４は、ＲＭＧヘッダのデータ構造を示す。
【００５２】
表５は、ＲＭＧデータのデータ構造を示す。
【００５３】

表６は、ｂｃｌｎｃｒモードからｂｃｌｎｃｒ値を得るためのルックアップ表である。

【００５４】

表７は、再符号化の繰り返し状態を示す。

【００５５】

表８は、他のメッセージホール処理モードを示す。

【００５６】

発明の詳細な説明

１．概要

この発明は、仮想的に不可視なマークの圧縮されたデータストリームへの付加を提供する。このマークはＤＶＤなどの情報記録媒体の公認されていないコピーのソースの追跡を可能にする「ソースオブコピーイングメッセージ」を構成する。本発明の最良な形態はＭＰＥＧの分野で生成されるビデオストリームの内容の下に述べられるが、本発明はこれに限定されるものではない。例えば、ソースオブコピーイングメッセージは、同様の原理を用いるオーディオ信号やソフトウェアにも組み合わせることができる。これらのマークは、ビューアでは見ることができず、オーソライズされた機関によって維持されるハードウェアによって検出され、デコードされる。より詳しくは、メッセージは特定の再生装置およびＤＶＤのような特定の原本記録媒体、および再生時刻のシリアルナンバーを特定する。明らかなように、このメッセージは再生装置によって媒体が再生されるごとに一意的に生成される。媒体（公認されたコピーから海賊版が作られたものであるとしても）を再生し、ビデオストリームに付加されたマークに埋めこまれたメッセージをデコードすることによって、機関は複製者の追跡を開始することができる。これらのマークの内容は再生されるたびに変化するので、このマークは「ランニングマーク」またはＲＭと称され、著作の時点で一旦作成されたら変化しないすかしとは異なる。

【００５７】

図１を参照して概要で述べられるのは本発明のランニングマークシステムが実施される環境である。記録スタジオでは、ムービーなどのプログラム素材を運ぶ

マスターディジタルリニアテープＤＬＴが、ユニット１２４によって、ＤＬＴの内容をＭＰＥＧビデオストリームに変換する処理がなされる。次に、プログラム素材は、ＤＶＤマルチプレクサユニット１２６によってＤＶＤ記録に適したフォーマットに配置される。エンコーダ１２４およびマルチプレクサ１２６は、この業界で標準的な装置を含んでいる。ＤＶＤマルチプレクサ１２６は、ＭＰＥＧビデオストリームに、データエントリのためにビットストリームに空領域を予約するサブピクチャ箇所保持部を付加する。

【００５８】

マルチプレクサユニット１２６の出力においてビデオストリームによって表されるイメージは、ディスクフォーマッティングユニット１２８によって、ビデオストリームのフレームの中でランニングマークの位置の候補を特定するために処理され、ビデオを保持するＤＶＤは、ユニット１３０でスタンプされ、シリアライズされる。ディスクフォーマッティングユニット１２８は、マークが見えないかまたはじゃまにならない画像のロケーションにランニングマークを埋めこむために、各ビデオフレームから候補ブロックまたはマクロブロックを探す。ＭＰＥＧビデオストリームのこれらのロケーションで「メッセージホール」（メッセージホールは、本発明では、それぞれビットモジュレーションに変換され得るピクチャフレームのブロックまたはマクロブロックに対応するＭＰＥＧセグメントである。）を表すデータビットには、フレーム中のロケーションを特定するためのアドレスオフセットが与えられる。置換されたブロックは、ディスクフォーマッティングユニット１２８によって、プレースホルダに保存される。

【００５９】

ユニット１２８は、ランニングマーク保持ユニット１３２（好ましくはテープまたはＤＶＤ）に、オリジナルのＭＰＥＧストリームだけでなく、再生装置、ディスク、および再生時刻のためのビデオストリーム情報を運ぶメッセージホールロケーションを送る。保持ユニット１３２は、複製者が追跡されるのを可能にするのを助けるために、テストの下の記録媒体からランニングマークを抽出しデコードするのに用いられるために、公認された機関に参照媒体として保持される。

【００６０】

ユーザは、機関から発行されたシリアルナンバーによって特定されるプレイバックユニット１３４が提供される。プレイバックユニットが支給されたユーザは、機関に登録される。また、登録ユーザによって閲覧されるためにムービーを運ぶ事前に記録された媒体も、機関によって支給されシリアライズされる。プレイバックユニットおよび事前に記録された媒体の登録ナンバーは、不可視であるがプレイバックユニット１３４によって読み取り可能に保存される。例えば、プレイバックユニットのシリアルナンバーは、ユニット内のリードオンリーメモリに保存され、媒体のシリアルナンバーは、好ましくは媒体に記録されているものの中にエンコードされる。プレイバックユニット１３４は、都合よくは媒体が再生される時刻を特定するリアルタイムクロックを含む。これらの情報コンポーネントは、媒体が再生されるごとに、プレイバックユニット１３４によって、シリアライズされ、生成されるランニングマーク中にエンコードされる。

【００６１】

登録されたプレイバックユニット１３４は、事前に記録された媒体の内容を再生し、通常は、ビデオストリームがテレビ画面やモニタ１３５に再現される。一例は、事前に記録されたムービーを見る目的でＤＶＤを借り受けまたは購入するカスタマである。ビデオストリームに埋めこまれたランニングマークは、モニタ１３５上では見ることはできず、ビューアは、ランニングマークを関知したり影響を受けたりしない。しかし、もし媒体がリプロダクションユニット１３６を用いてコピーされた場合には、そのようにして作成された公認されていないコピー１３８は、オリジナルのビデオストリーム（ムービー）だけでなく、プレイバックユニット１３４、機関から発行されたオリジナルディスク、および再生時刻を特定するランニングマークをも含む。公認されていないコピーは、マーク抽出システム１２２によって再生され、複製者を追跡するために好ましくは機関によって保持される。マーク抽出システム１２２は、マーク保持システム１３２から得られたブロックロケーションでのランニングマークの内容を読み取る。ランニングマークの内容の適切なデコーディングは、公認されていない複製の出所の追跡に用いられる。

【００６２】

ランニングマークは、好ましくは選択されたフレーム内のビデオストリーム中に戦略的に分配されたピクセルのブロックを含むが、ランニングマークは、ビューアによる関知を阻止するために、連続するフレームないの同じロケーションには現れない。フレーム中のランニングマークの位置は、プレースホルダセクタによって定義されたメッセージホールロケーションに対応する。マーク自信は、メッセージホールロケーションのピクセルがオリジナル画像の内容から変更されるか否かによらず、“１”または“０”の論理で表されてもよい。この基準は、反転され、または他の異なるフォーマット、例えばバイナリデータではなく４値などにエンコードされ得る。一例としては、もし、メッセージホールが、ブロックのオリジナルの画像内容から変更されたピクセルのブロックを含む場合には、そのメッセージホールは論理“１”としてデコードされ、もしメッセージホールロケーションに対応するピクセルブロックが、そのロケーションのオリジナル画像と比較して変更されていない場合にはメッセージのないようは論理“０”としてデコードされてもよい。これらの論理１および０は、フレーム内で、そして、フレームごとに、コピーのソースに対応する完全なメッセージを組み立てるために、連続して読み込まれる。

【００６３】

図２は、オペレータによる手動で、または既に述べた画像属性の基準に従った装置によって自動的に、ビューアに対しては不可視にフレーム中に配置されたメッセージホールロケーションａ、ｂ、およびｃを含むあるビデオフレームに対応する画像を表す。例えば示された画像の境界などの変わり目、または加工された領域は、選択に好ましい。もし、ブロックａの内容がオリジナルの画像から変更されると、それは値が“１”のランニングマークとして、さもなくば、もしフレーム中のマークａがオリジナル画像と同一である場合には値が“０”の値のランニングマークとしてデコードされる。ここで重要なことは、メッセージホールのロケーションは保存され、ランニングマークを表すそれらのロケーションのピクセルブロックの内容は、評価のために、ランニングマークが変更された画像を表すか否かを決定するために同じロケーションのオリジナル画像と比較される。しかしながら、他のエンコード基準、例えば、バイナリまたは他のタイプのデータ

を指定するための２またはそれ以上の異なって変化したランニングマークを使うことも、ランニングマークの値を決定するのに用い得ることは理解されるべきである。

【００６４】

図３は、ＭＰＥＧビットストリーム中のメッセージホールを表す。いくつかの参照ロケーションに関しての各メッセージホール、ビットサイズ、およびロケーションオフセットは、記録媒体のサブピクチャチャネルに保存される。複製のソースに対応するＭＰＥＧビットは、プリエンコードされ、サブピクチャチャネル内に保存される。プレイバックユニット１３４でのランニングマーク（ＲＭ）データのエンコーディング過程では、プレイバックユニットおよび媒体のシリアルナンバー、ならびに再生時刻が読み出され、入力されるＭＰＥＧビットストリームのメッセージホール中のビットが、この複製情報のソースに対応するＭＰＥＧビットに置き換えられる。いかなる数のビットも、機関によって発行されたプレイバックユニットおよび媒体の数に適応して適切に定義された範囲で、各メッセージエレメントに対して指定されることができる。これらのエレメントに対応したビットは、媒体が再生されるごとに再生ソースを特定するビットストリームを形成するためにシリアライズされる。

【００６５】

２．ランニングマークシステム

上記概要のように、ランニングマークを用いてビデオストリームに埋めこまれたメッセージは、好ましくは１またはそれ以上の次のフィールド（１）プレイバックユニットのシリアルナンバー、（２）記録媒体（例えばＤＶＤ）のシリアルナンバー、および（３）ソースの複製を示す再生時刻を含む。各ランニングマークメッセージは規定された数、例えば、連結され、複製のソースオブコピーイング特定フィールドを含む場合には、１２８のビットから成り立っている。

【００６６】

特定のためのソースオブアイデンティフィケーションメッセージがエンコードされ、ＭＰＥＧビデオストリームに挿入される方法を図４ａに示す。記録媒体が、閲覧されるため（または公認されていない複製のため）に再生される際に、プ

レイバックユニット１３４（図１）でリアルタイムに生成されるメッセージ、図４ａにおいてはＭｓｇｌｎと称される、は、メッセージエンコーダ３０２で適用される。このエンコーダ３０２は、メッセージの全てのフィールドを各フィールド（プレイバックユニットシリアルナンバー、ディスクシリアルナンバー、再生時刻）に割り当てられたビットの数にしたがってビットストリームＢａに変換する。プレイバックユニットのシリアルナンバーは変更されないので、好ましくはプレイバックユニット内のリードオンリーメモリ（ＲＯＭ）から読み出される。ディスクのシリアルナンバーは、再生中に直接ディスクから読み込まれる。再生時刻は、プレイバックユニット内のシステムクロックから得られる。全てのビットは、ビットストリームＢａを形成するために連結される。

【００６７】

好ましくは複製のソースオブコピーイングメッセージは、ＤＶＤまたは他の媒体のビデオストリームに複数回連続して挿入されてもよい。これは、ＤＶＤの中でメッセージデータに利用可能なフレームの数が、コピーソースを特定するために必要なデータの記憶要求よりもはるかに大きいために可能となる。メッセージの安全性を高めるのを助けるためには、フォーマットまたは各挿入の内容を異ならせてもよい。例えば、連続するいくつかの各エントリのコーディングや、同じ特定を表す全ての情報は、機関のデコーダによって理解される異なるフォーマットで格納されてもよい。

【００６８】

最良の形態では、唯一のデータのビットが各メッセージホールに埋めこまれる。しかし、発明の範囲内で種々の変形が可能である。例えば、メッセージの２またはそれ以上のビットが、何らかの適切な方法でエンコードされて一つのメッセージホールにもたらされてもよい。また、１つのメッセージビットが２またはそれ以上のメッセージホールに分配されてもよい。

【００６９】

さらに他の変形例としては、ランニングマークは、ＭＰＥＧビットストリーム中よりもビデオストリーム中に直接埋めこまれてもよい。すなわち、フレームの各ピクセルまたは選択されたいくつかのピクセルが、直接、メッセージビットと

ともにエンコードされてもよい。これは、例えば、オリジナルに関してピクセルの輝度を変化させて、所定の論理値を示すようにしてもよい。この技術の現実的な実施は、論理値１または０をそれぞれ表すために、各輝度が増加または減少されるピクセルによって示されるデータのＰＮシーケンスによって提供される。

【００７０】

２．１　メッセージの暗号化

ビットストリームＢａが暗号化ユニット３０４に供給される。暗号化ユニット３０４では、再生源となるメッセージを保護するため、また、偽造されたものであるけれども適法なメッセージを攻撃者が作成してシステムの信用を落とすのを阻むために強固な暗号化が行われる。メッセージのビットストリームＢａの暗号化には、図５に示すような５重の暗号化が適用されることが好ましい。図５に示す５重の暗号化ではＤＥＳ（Ｄａｔａ　Ｅｎｃｒｙｐｔｉｏｎ　Ｓｔａｎｄａｒｄ）アルゴリズムを使用している。５重のＤＥＳアルゴリズムの前と後には１２８ビットの置換が行われる。まず、１２８ビット置換器Ｐｓ０　３２０によってビットストリームＢａがスクランブルされ、スクランブルされた結果が左半分の部分Ｂａｌと右半分の部分Ｂａｒとに分割される。

【００７１】

左半分・右半分の各部分は６４ビットで構成される。左半分・右半分の各部分はそれぞれ、ＤＥＳによって鍵Ｋ０－Ｋ４を用いて５回暗号化される。鍵Ｋ０－Ｋ４の各々は６４ビットの長さを有する。その後、置換器Ｐｓ１　３２２によって、左半分の部分と右半分の部分とが結合され、結合された結果がスクランブルされ、１２８ビットのビットストリームＢｂが生成され、これが出力される。

【００７２】

ＤＥＳに代わる暗号化の手段としては、公開鍵暗号系がある。公開鍵暗合系は、整数論、有限体構造および代数的な符号理論に基づいている。公開鍵暗合系の例としては、ＲＳＡ、ＭｃＥｌｉｃｅ、ＥｌＧａｍａｌ、楕円曲線アルゴリズムなどがある。ＩＤＥＡ（Ｉｎｔｅｒｎａｔｉｏｎａｌ　Ｄａｔａ　Ｅｎｃｒｙｐｔｉｏｎ　Ａｌｇｏｒｉｔｈｍ）と呼ばれる新しい暗号アルゴリズムも暗号化手段のよい候補である。ＩＤＥＡは、１９９２年にＸｕｅｊｕａ　ＬａｉとＪａｍ

ｅｓ　Ｍａｓｓｅｙとによって提案された暗号化アルゴリズムである。

【００７３】

２．２　誤り訂正

暗号化ユニット３０４から出力された１２８ビットのビットストリームＢｂは、テープまたはディスク再生技術において不可避なひずみを訂正するために誤り訂正符号化器３０６に供給される。ひずみは、デジタルーアナログ変換処理、ソースレコーディング、ディスクの製造、テープまたはディスクの複製などの過程で生じる。模倣者は、ビデオソース（ｖｉｄｅｏ　ｓｏｕｒｃｅ）を故意にひずませることによって本発明のランニングマークシステム（ｔｈｅ　ｒｕｎｎｉｎｇ　ｍａｒｋ　ｓｙｓｔｅｍ）を攻撃するかもしれない。埋め込まれたランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）を確実に回復することができるようにするために、強固な誤り訂正符号化（ＥＣＣ）が行われることが好ましい。さまざまなＥＣＣアルゴリズムおよびそれらの組合せを使用することができる。高い誤り訂正能力を達成しかつ少ない計算量で符号化するためには、図６ａおよび図６ｂに示すような連接ＥＣＣシステムを適用することが望ましい。このＥＣＣシステムでは、８ビットシンボルの（３２，１６）縮小リード・ソロモン符号化器によって１２８ビットの入力ビットストリームＢｂが符号化されて２５６ビットのビットストリングＢｃ１が得られる。得られたビットストリームは、図６ａに示すように符号化率１／２、拘束長Ｋ＝７の畳み込み符号化器によって符号化されて５２８ビットの出力ビットストリームＢｃが生成される。この符号化処理では３２シンボルのうち１つの符号語だけしか存在しないため、速度が速く、符号化器の間でビットインタリーブが必要とされない。

【００７４】

リード・ソロモン（ＲＳ）符号は一般化ＢＣＨ符号のある特別のサブクラスである。ガロア体ＧＦ（2^m）において定義された通常の（ｎ，ｋ）ＲＳ符号は、長さｎ（＝2^m-1）の符号語のシンボルを有する。ここで、ｍは正の整数である。各シンボルはｍビットで構成される。この符号の最小距離は（ｎ－ｋ＋１）であり、この符号は、最大限ｔ＝〔（ｎ－ｋ）／２〕個の誤りまで訂正できる。ただし、〔ｘ〕は、ｘ以下の最大整数を示す。パリティ検査シンボルの数は（ｎ

－ｋ）である。この符号の生成多項式ｇ（ｘ）は以下の示すとおりである。

【００７５】

$g(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^3)\cdots(x-\alpha^{21})$

αはＧＦ（2^m）の原始元である。ｍ＝（m_0，m_1，・・・，m_{k-1}）は、一致多項式ｍ（ｘ）を伴うｋシンボルのメッセージブロックであるとする。ここで、$m(x)=m_0+m_1x+m_{k-1}x^{k-1}$である。また、ｃ（ｘ）＝（$C_0$，$C_1$，・・・$C_{n-1}$）は、一致多項式ｃ（ｘ）を伴うｎシンボルのＲＳ符号である。そして、ある（組織）ＲＳ符号化アルゴリズムは以下の２つのステップからなる。

【００７６】

ステップ１：ｍ（ｘ）とx^{n-k}との積をｇ（ｘ）で割る。ｄ（ｘ）＝$d_0+d_1X+d_{n-k-1}x^{n-k-1}$を別途記憶しておく。

ステップ２：Ｃ（ｘ）＝$x^{n-k}m(x)-d(x)$をセットする。ここで、ｃ＝（$-d_0$，$-d_1$，・・・，$-d_{n-k-1}$，m_0，m_1，・・・，m_{k-1}）である。

【００７７】

上述の多項式の計算はＧＦ（2^m）内で行われる。シフトレジスタを用いることによって効果的に行うことができる。ＣＤやＤＶＤプレーヤの多くのアプリケーションにＥＣＣが含まれている。許容される符号長は、通常の長さであるｎ（＝２ｍ－１）シンボルよりも短いため、縮小ＲＳ符号を用いることができる。Ｓは、（ｎ，ｋ）ＲＳ符号の符号語のサブセットであるものとする。（ｎ，ｋ）ＲＳ符号の符号語中もっとも右のｊ座標は値０を有する。そうすると、Ｓ内のすべての語からもっとも右のｊ座標を削除することによって（ｎ－ｊ，ｋ－ｊ）縮小ＲＳ符号が形成される。元のＲＳ符号を生成する同様のシフトレジスタ符号化器を、短縮ＲＳ符号を生成するために使用することができる。もし削除された座標を消去された位置として取り扱ったならば、オリジナルのシフトレジスタ符号化器と同じものを消去したものの復号化とともに復号のために使用することができる。（消去したものとは、受信したシンボルの値が疑わしいことを示すシンボルである。）これに代えて、この発明では、削除された座標をゼロで置き換える手法を使用する。

【００７８】

この発明のＲＭシステムでは、１２８ビットのメッセージを符号化するためにＧＦ（2^8）における短縮ＲＳ（３２，１６）符号が使用される。各シンボルは８ビットの長さを有する。この符号では最大ｔ＝８のシンボルの誤りまで訂正することができる。原始元αは、原始多項式Ｐ（ｘ）の根である。ここで、Ｐ（ｘ）＝$1+x^2+x^3+x^4+x^8$　である。ＲＳ符号を効果的に復号するために、Ｂｅｒｌｅｋａｍｐ－Ｍａｓｓｅｙシフトレジスタアルゴリズムを使用することができる。

【００７９】

畳み込み符号化器は入力ビットストリームを多数のインパルス応答とともに畳み込んでＥＣＣビットを生成する。　図７は、符号化率１／２、拘束長Ｋの畳み込み符号化器を示す図である。ここで、ｘ＝（x_0，x_1，x_2，・・・）は入力ビットストリーム、（$g^{(0)}$，$g^{(1)}$）は符号化器に対する２つのインパルス応答であるとする。そうすると、符号化されたビットストリームｙ＝（$y_0^{(0)}$，$y_0^{(1)}$，$y_1^{(0)}$，$y_1^{(1)}$，$y_2^{(0)}$，$y_2^{(1)}$，・・・）が以下に示す離散的な畳み込みによって得られる。

【００８０】

$$y_i(j)=\sum_{l=0}^{K-1} x_{i-l}g_l^{(j)}$$

ここで、ｊ＝０、１である。畳み込み符合を復号するために、ビタビ復号化器を使用することができる。連接誤り訂正システムのために、畳み込み符号化器は拘束長Ｋ＝７および符号化率１／２を有する。インパルス応答は、$g^{(0)}$＝（１０１１０１１）および$g^{(1)}$＝（１１１１００１）である。

【００８１】

２．３　スペクトル拡散符号化／ＣＤＭＡ

多数の再生ユニットからのビデオ出力を結合することによってこの発明のランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）システムを攻撃することを模倣者は試みるかもしれない。この方法では、埋め込まれたランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）は削除され、コピー元のメッセージを回復させることが困難

になるかもしれない。このため、この発明の１つの局面によると、符号分割多元接続（ＣＤＭＡ）符号化ユニット３０８を設けて、再生ユニットによって生成されたランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）の復号化の分離をたとえノイズが存在しているときであっても可能にしている。ＣＤＭＡは、指定された周波数内で多数の伝送チャンネルを供給するために特別の符号を使用する変調の一種である。Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　Ｊｏｕｒｎａｌ，Ｄｅｃｅｍｂｅｒ　１９９３のページ９０－９７「Ｎｏｒｔｈ　Ａｍｅｒｉｃａｎ　Ｃｅｌｌｕｌａｒ　ＣＤＭＡ」というタイトルの記事にＣＤＭＡの標準について述べられている。以下の説明はこれを参考にしている。ランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）を符号化するためにＣＤＭＡを使用することの利点は、特性に基づいている。種々の再生ユニットは種々の相関が低い波形を用いてビットを符号化してＭＰＥＧビデオストリームに加えるという特性に基づいている。復号の間、受信信号と関連波形との内積をとることによって再生ユニットの送出ビットは回復される。これらの波形の間の相関は低いため、たとえ種々の再生ユニットからの出力波形の結合として受信波形が生成されていても内積をとることによって送出ビットを直接取り出すことができる。

【００８２】

この発明の１つの局面に従うと、ＣＤＭＡ符号化においては擬似ランダム雑音（ＰＮ）系列が使用されることが望ましい。これらＰＮ系列はスペクトル拡散多重アクセス用アプリケーションに使用されるものであり、時間領域においてランダム雑音のように振る舞う。また、周波数領域において信号を隠したり干渉を除去したりする特性が良好である。使用することができる別の種類の波形としては、バイナリーの直交波形の集合がある。直交波形が有効なのは、それらの相関が最も低いすなわち０であり、よりよくビットを取り出せるからである。この目的のためのバイナリー直交波形の望ましいタイプはＨａｄｍａｒｄ－Ｗａｌｓｈ（ＨＷ）波形である。これはＨＷマトリクスの行を周知の方法で選択することによって得られる。

【００８３】

しかし、非常に多数の再生ユニットには独自の波形が割り当てられる必要があ

るであろう。もし、各再生ユニットが異なった波形を使用したならば、要求されるHW波形の長さは非実用的なものとなりる。また、未知の波形でビットを取り出すのは不可能という問題がある。もし多数の再生ユニットが共通の波形を共有したならば、模倣者はこれらの再生ユニットを使用してランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）を回復できないようなビデオソースを生成するかもしれない。この問題を解決するためにこの発明では、ランタイムの間、各ディスクプレーヤーに対して１つの波形を波形の集合からランダムに選択する。これは有効である。なぜなら、模倣者のすべてのディスクプレーヤーがランタイムの間同じ波形を選ぶ確率は低いからである。また、映像内のランニングマーク（ｒｕｎｎｉｎｇ　ｍａｒｋ）の各コピーについて波形はランダムに選ばれる。そして各時間の間、同じまたは異なった映像が再生される。少数の異なる波形だけがあるため、ビットの取り出しプロセスではそれらのすべてを試して内積の最も高いものを選択することによって送出ビットを決めることができる。高速HW変換をこの目的のために利用することができる。

【００８４】

もしどの波形を使用するかが決まっていれば、ＣＤＭＡシステムに対する模倣者の攻撃を避けるために、ＣＤＭＡ波形は置換を使用してスクランブルされ、ビットストリームを送出するために１つの波形よりも多くの波形が使用される。直交特性を保存するために、波形はまず異なる列の間で置換され次いで異なる行の間で置換される。表１は、長さ８のＨａｎｄｍａｒｄ－Ｗａｌｓｈ波形を示す。表２は、列および行の置換後のＨａｎｄｍａｒｄ－Ｗａｌｓｈ波形を示す。

【００８５】

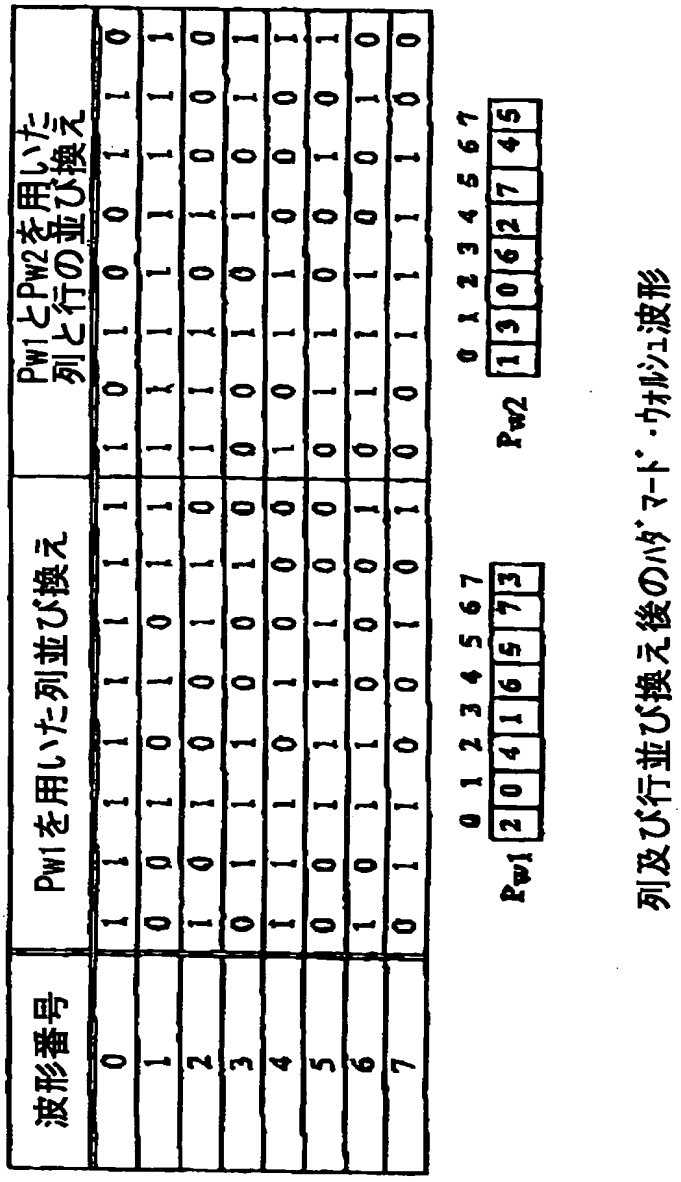
【表１】

n=8の時のﾊﾀﾞﾏｰﾄﾞ・ｳｫﾙｼｭ波形

波形番号	値								2値再表現							
0	+1	+1	+1	+1	+1	+1	+1	+1	1	1	1	1	1	1	1	1
1	+1	+1	+1	+1	-1	-1	-1	-1	1	1	1	1	0	0	0	0
2	+1	+1	-1	-1	+1	+1	-1	-1	1	1	0	0	1	1	0	0
3	+1	+1	-1	-1	-1	-1	+1	+1	1	1	0	0	0	0	1	1
4	+1	-1	+1	-1	+1	-1	+1	-1	1	0	1	0	1	0	1	0
5	+1	-1	+1	-1	-1	+1	-1	+1	1	0	1	0	0	1	0	1
6	+1	+1	-1	-1	-1	-1	+1	+1	1	1	0	0	0	0	1	1
7	+1	-1	-1	+1	-1	+1	+1	-1	1	0	0	1	0	1	1	0

【００８６】

【表２】



波形番号	Pw1を用いた列並び換え	Pw1とPw2を用いた列と行の並び換え
0	1 1 1 1 1 1 1 1	1 0 1 0 0 1 1 0
1	0 0 1 0 1 1 1 1	1 1 1 1 1 1 1 1
2	1 0 1 0 0 1 1 0	1 1 1 1 0 1 0 0
3	0 1 1 1 0 0 1 0	0 0 1 0 1 0 1 1
4	1 1 1 0 1 0 0 0	1 0 1 1 0 0 0 1
5	0 0 1 1 1 1 0 0	0 1 1 1 0 0 1 0
6	1 0 1 1 1 0 0 1	0 1 1 1 1 0 0 1
7	0 1 1 1 0 0 1 0	0 0 1 1 1 1 0 0

Pw1 0 1 2 3 4 5 6 7
2 0 4 1 6 5 7 3

Pw2 0 1 2 3 4 5 6 7
1 3 0 6 2 7 4 5

列及び行並び換え後のﾊﾀﾞﾏｰﾄﾞ・ｳｫﾙｼｭ波形

【００８７】

スクランブルされたHW波形は、あらかじめ符号化されたディスクから読み出されまたはあらかじめＲＯＭに記憶されている。すなわち、ランタイム中に計算する必要がない。多数の波形を使用するためには、それらはビットストリームの同じビット位置について異なった再生ユニットに対する波形は異なっているように適切に選択されるべきである。

【００８８】

図８を参照してＣＤＭＡ符号化処理をより分かりやすく示す。まず、０から５１１の間の数がランダムに生成される。そして、列と行の置換を伴った２６４の連続的なHW波形が選択される。これらの波形とビットストリームＢｃの最初の半分の部分および２番目の半分の部分との排他的論理和のビット演算が行われ、５２８×５１２の長さの出力ビットストリームＢｄが生成される。バーストノイズをさけるために、ビットストリームＢｄのビットは列の順に形成される。

【００８９】

２．４　スクランブル

より高い安全性をビットの送出に持たせるために、ＭＶＬＰ（ｍｕｌｔｉｐｌｅ　ｖａｒｉａｂｌｅ　ｌｅｎｇｔｈ　ｐｅｒｍｕｔａｔｉｏｎ）置換が使用されてＣＤＭＡビットストリームがスクランブルされる。図９ａを参照して、置換器３１０において、種々の長さを有する多数の置換器を使用してＣＤＭＡビットストリームＢｄのビットが置換されて同じ長さのビットストリームＢａが形成される。図９ｂに示すように、最大長２５６の８個の置換器を使用することができる。長さ７の置換器Ｐ０およびその逆置換器の例を図９ｃに示す。最終の置換を行うために、残っているビットの数は置換器の数よりも少ない。ギャップを満たすために”Ｄｏｎ’ｔ　ｃａｒｅ”の語が詰め込まれる。置換の後、正常なビットは結果からスキャンされて送出される。逆置換処理の間、ユニット４０６のデコーダ４００において、送出ビットが正しい位置に挿入された後、逆置換が実行される。図９ｄおよび図９ｅは、図９ｃに示した置換器を用いたこれらの処理の例を示す。上記置換は、上述したＣＤＭＡ符号化処理と結合することができる。ビットストリームＢｅを送出する前に必要となる一時的な記憶スペースは、５２８×５１２ビットの代わりに２５６ビットである。置換されたビットストリームＢｅはビットモジューラ３１２に供給される。ビットモジューラ３１２は、入力されるＭＰＥＧビットストリームを修正することによって１またはそれ以上の送られてきたビットを最終ビデオ出力に埋め込む。デコーダ（図４ｂ）内のビット復調器４０４において受信ビデオを加工することによってこれらのビットは回復される。

【００９０】

２．５　ビット変調及び復調

図４ａに示すように、ビット変調器３１２は、入力ＭＰＥＧビットストリームのメッセージホールのビットストリームを変更することによってビットストリームＢｅの１個以上の送信されたビットを修正ＭＰＥＧビットストリームＢｆに埋め込む。このビットストリームＢｆはビデオ信号を含む走行マークを出力するＭＰＥＧ復調器３１４に付与される。各メッセージホールについて、そのサイズとある基準位置に対してオフセットされた位置とがＤＶＤディスクの部分画像チャネルに格納される。ＲＭキャリアは送信用ＲＭビットからなる（ＭＢの画素値にも対応する）ＭＨ内のＭＰＥＧビットとして定義される。これらキャリアのＭＰＥＧビットは予め符号化され、部分画像チャネル内に格納される。ＲＭ符号化時、ＭＨ情報とこれらキャリアとが読み出され、入力ＭＰＥＧビットストリームのメッセージホール内のビットが送信用ビットに応じて上記キャリアに置換される。これらキャリアを格納する必要なデータ空間を削減するために、キャリアの一つがＭＰＥＧビットストリームのＭＨに予め格納される。ＭＨは、その内部に２のｎ乗個の異なるＲＭキャリアが存在する場合に、ｎビットを発信することができる。最良の形態では、各ＭＨにつき２個のキャリアが存在し、１個のキャリアのみが部分画像チャネルに格納される。

【００９１】

図４ｂのビット復調器４０４は受け取ったビデオ信号を処理することでビットストリームＢｅを復元する。受け取られるビデオ信号はＤＶＤ再生装置などの他のソースや、ＶＨＳテープ再生装置やインターネットから得ることができる。走行マークを復号化する前に、ビデオディジタイザ装置４０２でＮＴＳＣ復調などの適正な復調とフォーマット変換が実行される。ビデオディジタイザ４０２は高品質のＡ／Ｄ変換器を使用して入力ビデオ信号をディジタル化する。チャネル補償器は、ディジタル化されたビデオ信号を元のビデオ信号に合わせ、従来のフレーム位置合わせ技術とチャネル等化技術とを使用してチャネルの誤りを補償する。

【００９２】

さらに図４ｂに示すように、ビット復調器４０４は、ビデオフレーム内に配分

されたメッセージホールの元のビデオ信号とそのブロック位置またはマクロブロック位置とともに、ディジタル化されたビデオビットストリームＢｂを受け取る。ビット復調器４０４は、ディジタル化されたビデオストリームＶｂと元のビデオ（基準となるビデオと検査対象ビデオ）が同時に再生されている際にこれら二つのビデオを比較し、基準ビデオに記録された各メッセージホール位置により定義された検査対象ビデオのブロックまたはマクロブロックの内容を比較する。メッセージの位置に関して、元のビデオ（基準ビデオ）のブロックまたはマクロブロックと検査対象ビデオ内のブロックまたはマクロブロックが画像の変化を示していれば、ビット復調器は論理値１を生成し、元のビデオのブロックまたはマクロブロックと検査対象ビデオ内のメッセージホール位置のブロックまたはマクロブロックが同じであれば、ビット復調器４０４はビットストリームＢｅとして論理値０を生成する。このビットストリームＢｅは、復号器４１４でコピー源メッセージに変換される前に、ユニット４０６でデペルミュレート化（スクランブル解除）され、ユニット４０８で復号化され、ユニット４１０で誤り訂正され、ユニット４１２で暗号解除される。ユニット４０６，４０８，４１０及び４１２はそれぞれ図８の対応部３１０，３０８，３０６及び３０４の逆の機能を実行する。

【００９３】

３．ＲＭキャリアの生成

ＲＭキャリアは、ふつうの人は見ることがほとんどできないように、検出されたり取り除かれたりしないように、また、送信メッセージを再生することが十分に確実にできるように、慎重に設計する必要がある。ＲＭキャリア及びMH情報を格納するために使用することができるスペースは限られている。このことは、利用できる送信ビットの総数及びＲＭメッセージの再生性能に影響する。この他に、ＲＭキャリアを含めたＭＰＥＧのビットの総数は、いくつかのレベルにおいてムービーポストハウス（movie post house）からの元のＤＶＤビットストリームのビット数の総数に等しくなければならないという制約がある。そうでなければ、ＭＰＥＧ復号化におけるバッファオーバーフローやアンダーフローの問題が生じる可能性があり、ナビゲーション（navigation）及び元のＤＶＤディスクの

他のＤＶＤ情報を変更しないでおくために、ＤＶＤの様々なセクタのビットストリームを編集することが必要となる。

【００９４】

ビットの総数を変更しないでおくレベルはいろいろあり、これがキャリアの設計に影響を与える。最も簡単なレベルでは、１つ又は少数のビットを変更する。例えば、動きベクトルやＤＣＴ係数の符号ビットを変更したり、量子化スケールファクターを変更する。他のレベルでは、ＭＢのビット総数を変更しないようにする。例えば、ＤＣＴ係数を同じビット数のもので置き換えたり、ビット総数を変えずに複数のＤＣＴ係数を変更する。これらの２つの方法の主な有利な点は、一般にキャリアのサイズは小さいという点と、キャリアのための格納スペースが余分に必要ではない場合があるという点である。しかし、これらの方法には、大きくて制御することが難しい映像歪みがあり、メッセージ再生の可能性が低く、ＭＨに適したＭＢの数が少ないという傾向がある。キャリアのビット数は、元のＭＢのビット数よりも大きい（又は小さい）ことが望ましい。キャリアの大きさが元のものより小さい場合と同様に、ＭＨの全てのキャリアのサイズを同じにするために、ビット埋め込みが行われる。ＭＰＥＧビットストリームの変更を確実に合法的にするため、ＭＨのＭＢの後に新たなスライスヘッダが挿入される。同じ行のビット総数が変わらないように、その行の全てのＭＢを再符号化することによって、増加したビットの総数が補償される。再符号化の方法については、後により詳細に説明する。

【００９５】

ＭＢの周りの画素の強さを変えることによって、マクロブロックのエッジにおける変化を目に見えないように滑らかにする。例えば、図１０のように、画素ｐ（ｎ）の周りの画素ｐ（ｍ）には、遷移領域を作るために、歪みが少し与えられる。

【００９６】

本発明においては、ＲＭキャリアを生成する基本的な方法は、映像源の強さを加えたり変えることである。この方法は、

（１）映像にノイズを与える

（２）エッジに沿って強度を変える
（３）テクスチャ領域の画像の特性を変える
というステップを含む。また、所定のメッセージホール（message hole）におけるビットの状態が変化したことが１つのバイナリ値を示すようにし、ビットの状態が変化しなかったことが他のバイナリ値を示すようにしてもよい。動きベクトルの精度を１画素精度と半画素精度との間で切り換えてもよい。１つ以上の輝度ブロック又は色差ブロックのＤＣＴ係数を変更してもよい。更に、
（４）所定の論理値のメッセージビットに応じて、ＭＢの画像のエッジのプロファイルを修正する（図１０ｂは、第１及び第２の論理値を示し、増加又は現象するエッジのプロファイルを図示している）。
（５）図１０ｃのように、画像のテクスチャを１画素、又は１画素以上若しくは１画素以下だけシフトし、所定の論理値のデータを示す。
（６）マクロブロックのテクスチャを伸長又は圧縮して示す（図１０ｄ）。

【００９７】

このような方法や他の方法によってＭＢの変更し、メッセージのビットを定義することは、目に見えないように画像のテクスチャ部分において行うと有利である。同様に他の方法を採用することも可能である。重要な点は、変化して論理値を表すパラメータは、確実に検出されなければならない（ロバスト性がある）が、画像と混ざっていなければならないということである。

【００９８】

ランニングマーク（running mark）の変調をするために映像にノイズを付加することは、歪みが小さく、コピーをする人に検出されにくいという点で有利である。図１１に示すように、ノイズのようなキャリアは、同程度の歪みに対して、ＭＢ内で歪みを拡散する。このため、各画素の歪みは小さくなり、事実上目で見ることはできなくなる。カメラやフィルムからのノイズのように、ノイズは映像ソースに元から生じているので、ノイズのようなキャリアは、フレーム差分を求めても検出することは難しい。

【００９９】

本発明では、ＭＨの各ＤＣＴブロックの第１のｎＤＣＴ　ＡＣ係数を変更する

ＤＣＴによる方法がＲＭキャリアを生成するために用いられており、これは図１２に示されている。配列MHBlock[0][block]が、ＭＨ（ＲＭを除く）の元のＤＣＴブロックを全て格納するために用いられる。配列MHBlock[1][block]及びMHBlock[2][block]は、リプレースメント（replacement）１及び２を作るためにノイズを加えた後のブロックを格納するために用いられる。キャリア０及び１は、元のブロック及び２つのリプレースメントから選択することができる。まず、ｎＤＣＴの要素の配列がランダムに生成される。これらの要素は、０，１，及び２のいずれかの値を持つ。その割合は、例えば０が５０％、１及び２はともに２５％である。リプレースメント１を生成するために、要素が１及び２であれば、対応するＤＣＴ係数は固定値sigmaをそれぞれ加算及び減算する。リプレースメント１を生成する場合は、加算と減算とが逆になる。要素が０であれば、キャリアはともに元のＤＣＴの値を用いる。ｎＤＣＴ及びsigmaの値は、輝度ブロックと色差ブロックとで異なるものとなりうる。図１３は、この方法の例を示している。これは、ｎＤＣＴ＝４０，sigma＝５、０，１，２の比が２：１：１である場合の輝度ブロックの画素領域における典型的なノイズパターンである。

【０１００】

４．ＤＶＤプレーヤー環境におけるビット変調の実行

４．１　概要

ＤＶＤプレーヤー環境における図４ａのビット変調器３１２の実行について、以下に詳細に説明する。

図４ａにおけるＢｅは、総計Numbitビットであるとする。これらのビットは、ポインタBitsPtrで示される配列Bits[]に格納される。Bits[]の最後の配列要素が届くと、次に送るビットとして最初の要素が用いられる。映画には、ＲＭビットの１つ以上のコピーが埋め込まれている点に注意する必要がある。コピーの数は、ＤＶＤディスク内の利用できるＲＭスペースによって異なる。

【０１０１】

ＭＨの映像ストリーム、ＭＨのサイズと位置、処理パラメータを置き換えるためのデータは、ＤＶＤのＶＯＢＵ内のＲＭＧ＿ＰＣＫと呼ばれるセクタに格納される。ＲＭＧ＿ＰＣＫのデータレイアウトとデータ構造は、図１４、表３～５に

示されている。

【０１０２】

【表３】

DivxStreamId	アプリケーション
00000001b	ナビゲーションパック以外のＤＶＤパックのメッセージホールビットストリームを置換することにより送信されたＲＭメッセージ。いかなるナビゲーションパックも走行マーク処理用の変更は不可。
その他	他の Divx アプリケーション用に予約

様々な Divx アプリケーション用プロバイダ定義ストリームの DivxStreamId

【０１０３】

【表４】

変数	説明
u32 bc	RMG ビットアレイ Bits[ ]内の RMG_PCK に送られる最初のビットを指定
u8 NumRMG	RMG_PCK 内の走行マーク回数
u8 NumBS	ビデオストリーム置換用に使用する付加ビットストリーム数。少なくとも１個の付加ビットストリーム(NumBS>0)を有し、(NumBS+1)の値が２のべき数であること。
u8 Reserved A	将来の修正用に予約
u1 FNextRMG	FNextRMG=0 の時、全ＲＭＧデータ内に NextRMG データと予約済み領域は存在しない。FNextRMG=1 の時、全 NextRMG データが次のＲＭＧデータの配置用に使用される。
u1 FSizeHole	RMG_PCK 内のメッセージホールのサイズが変わる場合、ＲＭＧデータ内に SizeHole データが存在し、FSizeHole=1 である。それ以外は、FSizeHole=0 であり、SizeHole データはＲＭＧヘッダの後に発生する。
u6 ReplaceMode	メッセージホールのビットストリーム置換用に使用すべき適正なパラメータを有するアルゴリズムの決定

ＲＭＧヘッダのデータ構造

【０１０４】
【表５】

変数	説明
u16 NextRMG	RMG_PCK の始めから次の走行マークデータまでのバイトのオフセット。ゼロは連鎖の終わり。
u5 Reserved B	将来の修正用に予約
u11 SizeHole	走行マーク用メッセージホールのバイトサイズ
u32 SectorNum	メッセージホールが位置する V_PCK の論理セクタ数
u4 bcIncrMode	表６に記載の bc の増分を制御
u12 Offset	メッセージホールから SectorNum[ ]で指定された物理セクタの初めまでのバイトのオフセット。物理セクタ、パックヘッダ、パケットヘッダを含む。
u8 b[NumBS][SizeHole]	走行マーク１個の SizeHole 長さを有する付加ＭＰＥＧビットストリーム
Reserved C	将来の修正用に予約

ＲＭＧデータのデータ構造

【０１０５】
図１４ａにおいて、ＲＭＧ＿ＰＣＫは、このＲＭＧ＿ＰＣＫによって修正されるメッセージホールを含むＶ＿ＰＣＫのどれよりも物理的に前に格納される。これらのＶ＿ＰＣＫは、ＲＭＧ＿ＰＣＫが生じるＶＯＢＵの中にある必要はない。図１４ｂにおいて、ＲＭＧ＿ＰＣＫは、ＤＶＤ private_stream_1 パックであって、そのパケットは供給者が定義したストリームによってエンコードされている。パケットのstream_id及びsub_stream_idは、それぞれ"1011 1101b"及び"1111 1111b"である。これは、ＤＶＤビデオ仕様書（ＤＶＤ video specification）1.0のVi5-6ページの表5.1.1-1及び表5.1.1-2に記載されている。パックとパケットのヘッダは、ＤＶＤビデオ仕様書1.0のsection 5.2,Vi5-9に定義されている。DivxStreamIdは様々なDivxアプリケーションを示し、表３に定義されている。DivxStreamIdが"0000 0001b"である場合は、ランタイムモジュールは、本明細書に記載のアルゴリズム及びデータ構造に従って、ナビゲーションパックを除

いた全てのＤＶＤパックのメッセージホールのビットストリームを置き換える。ＲＭＧ＿ＰＣＫのその他の部分は、ＲＭＧヘッダ、予約されたＢ，SizeHole及びＲＭＧデータを有している。ＲＭＧ＿ＰＣＫの全てのデータは１つのセクタの中になければならない。すなわち、どのデータもセクタ境界を越えることはできない点に注意が必要である。

【０１０６】

図１４ｃ及び１４ｄは、それぞれＲＭヘッダ及びＲＭデータの様々な変数を示している。これらの変数の目的は、表４及び５にまとめられている。ＲＭヘッダはＲＭＧ＿ＰＣＫ内の全てのランニングマークによって用いられるＲＭアルゴリズム及びパラメータについてのグローバルな情報を提供する。NumRMGは、ＲＭＧ＿ＰＣＫ内のランニングマークの総数である。各ランニングマークは、１つのＲＭＧデータを用いる。変数ReplaceModeはメッセージホールのビットストリームの置き換えに使われる適切なパラメータとともに、アルゴリズムを決定する。

【０１０７】

例えば、図１５のように、ｂｃは、ＲＭＧビット配列Bit[]の中でＲＭＧ＿ＰＣＫに最初に送られるビットを示している。ＲＭが終了すると、ｂｃの値は次のＲＭのために次の式（他の状況においては、他の関係が適用される）、

ｂｃ ＝ ｂｃ＋bclncrValue　…（４．１）

に従って更新される。

【０１０８】

【表６】

bclncrMode		bclncrValue	公式
0	0000	1	bclncrMode+1
1	0001	2	
2	0010	3	
3	0011	4	
4	0100	5	
5	0101	6	
6	0110	7	
7	0111	8	
8	1000	0	8 - bclncrMode
9	1001	-1	
10	1010	-2	
11	1011	-3	
12	1100	-4	
13	1101	-5	
14	1110	-6	
15	1111	-7	

bclncrModeからbclncr値を得るﾙｯｸｱｯﾌﾟﾃｰﾌﾞﾙ
bclncr値は数式(1)におけるbcの更新増分値

【０１０９】

表６は、各ＲＭのbclncrModeから前記bclncrValueを決定するためのルックアップテーブルである。新たなＲＭＧ＿ＰＣＫのために、送信されるビットを再同期させるためにｂｃの値が用いられる。ＲＭＧビットはデータ型ｕ１６のBits[]に格納されているので、ｂｃによって示され、送信されたビットを得るためには、適切なビット抽出が必要である。ここで、ランニングマークのために送られるべきビットの数をn_0とする。これは、次の条件、

$2^{n0} \geqq (NumBS+1)$　　…（４．２）

を満たす最小の整数として定義される。

【０１１０】

NumBSはメッセージホールを置き換えるために用いられるＲＭデータ中のビッ

トストリームb[]の総数の増加分を示す。ＲＭデータには少なくとも１つのビットストリームデータが加わっている必要があり、（NumBS＋１）の値は２の累乗でなければならない。すなわち、NumBS ＞ ０、かつ、n_0 ＞ ０でなければならない。bcIncrValueとn_0とは値が異なっていてもよい。Ｖ＿ＰＣＫのメッセージホールの位置は、論理セクタ番号SectorNumとＲＭデータのｕ１２オフセットによって決定される。実行時のパフォーマンスをよくするために、ＲＭＧ＿ＰＣＫ内のセクタ番号は全て減少しないように並んでいるものと仮定する。オフセットは、メッセージホールとSectorNumによって示された物理セクタ（物理セクタ、パック、パケットヘッダを含む）の開始点との間のバイト数を示している。ＲＭＧ＿ＰＣＫ内のＲＭデータの追加ビットストリームの数は、ＲＭデータのオーバーヘッドを減らすため、変更しない。メッセージホールのサイズはSizeHoleによって決定される。ＲＭＧ＿ＰＣＫ内の全てのメッセージホールのサイズが変わらないと場合は、ＲＭＧヘッダのFSizeHoleフラグは０にセットされ、ＲＭＧヘッダの後には１つのReserved B及び１つのSizeHoleのみが送られる。その他の場合は、各ＲＭデータはReserved B及びSizeHoleを含む。この例では、１つのＲＭには１つのメッセージホールのみが処理される。

【０１１１】

実行時において、計算をすることなく、ＲＭデータの予約領域をスキップしてＲＭＧ＿ＰＣＫ内の次のＲＭデータへ直接跳ぶために、NextRMGが使われる。予約領域は将来の発展のために使われる。NextRMGは、ＲＭＧ＿ＰＣＫの始めからＲＭデータの始めまでのオフセットのバイト数である。最後のＲＭデータについては、NextRMGには０がセットされる。ＲＭヘッダのフラグFNextRMGが１にセットされると、各ＲＭデータはNextRMGを有し、これらのNextRMGの値は、次のＲＭＧデータへ行くために使われる。フラグFNextRMGが０にセットされると、各ＲＭＧデータには予約領域はなく、NextRMGが計算される。

【０１１２】

図１６のアルゴリズムは、実行時におけるランニングマークの処理を概念的に示したものである。実際の実施形態は種々の再生ユニットの制約によって異なる。図１６の３行目のRunTimeModule（）プロセスは、ＤＶＤデータストリームの

最後に達するまで（又は、他の割込みが発生するまで）、実行中はＶＯＢＵのデータパックを読み続ける。８行目のreadSector（）ルーチンは、ＶＯＢＵから次のデータパックを返し、９行目のreadFromSectorルーチンは、このデータパックの特定の変数の値を返す。データパックのストリームタイプは、変数Streamldによって決定される。データパックがビデオストリーム又はオーディオストリームのいずれか（すなわち、Ｖ＿ＰＣＫ、Ａ＿ＰＣＫ又は他のオーディオパック）であれば、復号化され、その結果は表示のためにビデオ又はオーディオポートに送られる。他の型のパックについては、対応する処理が進められる。ランニングマークプロセスは、ＲＭＧ＿ＰＣＫが読み込まれ、すなわち、このデータパックは供給者が定義した（private stream １の）ストリームであり、そのDivxStreamldは"0000 0001b"に等しい。その後、ＲＭＧヘッダが２６行目のように読み込まれる。メッセージホールのサイズが変わっていなければ、すなわち、FSizeHole＝０であれば、２８行目のようにSizeHoleが読み込まれる。このＲＭＧ＿ＰＣＫの全てのランニングマークは、１２行目のRunningMarking（）ルーチンを各ビデオセクタ毎に呼び出すことによって行われる。これらの処理においてエラーが生じた場合には、適切なエラー処理動作が開始される。

【０１１３】

４８行目のRunningMarking（）ルーチンは、現在のビデオパックを参照するＲＭデータを適用することによってランニングマークを行う。次のＲＭデータの位置のNextRMGは、ＲＭＧ－ＰＣＫから読み出され、又は計算によって求められる。これは、フラグFNextRMGの値による。FSizeHoleが１にセットされている場合は、SizeHoleフィールドが読み出される。その後、セクタ番号、ビットカウントインクリメントモード（bit count increment mode）、メッセージホールのオフセット、及び付加されたビットストリームb[]が読み出される。セクタ番号が現在のビデオセクタの番号に一致するときは、６６行目のReplaceMessageHoleData（）が呼び出され、ReplaceModeに従ってＭＨビットを置き換える。次のＲＭＧ＿ＤＡＴＡ構造に移る前に、ｂｃはbclncrModeを使って更新される。

【０１１４】

７４行目のReplaceMessageHoleData（）は、変数のモードに従って種々の方法

でMHビットの置き換えを行う。Mode＝０の場合は、MHビットはb[0]で置き換えられる。Modeが１又は２の場合は、図１５のように、ｂｃによって示されたBits[]からn_0ビットを抽出することによって、sendBitが作られる。Mode＝１の場合、MHビットはb[sendbit-1]で置き換えられる。version＝２の場合は、sendbitの下位n_0ビットが反転される。すなわち、n_0＝３の場合、"0000 0101"が"0000 0010"になる。その後、MHビットはb[sendbit-1]に置き換えられる。

【０１１５】

ランタイムモジュールのファームウェアのクラッシュを防ぎ、ＲＭＧ＿ＰＣＫのエラーに起因する映像の歪みを最小限にするため、実施の際には適切なチェックが必要である。誤りがあると、MHビットの置き換えが行われず、処理は次のランニングマークに移ってしまう。

【０１１６】

４．２　ランニングマーク位置ホルダーとランニングマークパック

ランニングマークパック（ここでは"Divx Packと称する）は、パケットが、供給者が定義したストリームとして符号化された、ＤＶＤ private＿stream＿１パックである。パケットのstream＿id及びsub＿stream＿idは、それぞれ"1011 1101b", "1111 1111b"に等しく、これらはＤＶＤビデオ仕様書1.0のVI5-6ページの表5.1.1-1及び表5.1.1-2に記載されている。sub＿stream＿idの直後のサブピクチャデータ領域には、"0x 4449 5658 312e 3023"（１６進数）に等しい６４ビットがある。これらのビットは、文字列"DIVX1.0#"を表している。パケットの残りのビットは０にセットされている。

【０１１７】

Divxパックは、Divxパック周波数において、ＤＶＤビットストリームに一様に分布している。これは次の式、

Ndivx ＝ truncate((MovieLength/(15*factor))＋0.5)

に従って決定される。

但し、ＮｄｉｖｘはＤｉｎｘパック間にはさまれるＭＰＥＧ符号化ビデオフレームの数である。ＭｏｖｉｅＬｅｎｇｔｈは、タイトル作成者によって特定された分単位での再生長さである。ＮＴＳＣの係数は１．０、ＰＡＬの係数は１．２で

ある。Ｎｄｉｖｘの計算値が２未満の場合、Ｎｄｉｖｘは２に設定する。Ｄｉｖｘパックは物理的に（一時的にではなく）ビデオパックに対して位置している。Ｎｄｉｖｘ符号化ビデオフレームの収集はＤｉｖｘビデオグループとして引用する。Ｄｉｖｘパックは、Ｄｉｖｘビデオグループの第１バイトを含む付随のビデオパックの前に物理的に起こる。その結果、ＤｉｖｘパックはＤＶＤの第１ビデオパックの前に現れる。Ｄｉｖｘパックの表示は一時的に付随ビデオパックと無関係となる。Ｄｉｖｘパックの表示時間は、付随Ｄｉｖｘビデオグループ前、又は後に一時的に起こる。

【０１１８】

Ｎｄｉｖｘの計算を示すため、８５分間のＮＴＳＣ映画を仮定する。公式から、Ｎｄｉｖｘは６である。したがって、Ｄｉｖｘパックが６符号化ビデオフレームごとにＤＶＤビットストリームに挿入され、Ｄｉｖｘビデオグループが６つのビデオフレームを含むことになる。各符号化ビデオフレームが１．２のビデオパックを取り、Ｄｉｖｘビデオグループが７．２のビデオパックからなると仮定する。Ｄｉｖｘパックのないビデオグループのレイアウトを図１７に示す。Ｄｉｖｘビデオグループ１からの最後のビデオフレームの部分が、ビデオグループ２からの第１ビデオフレームと同じパックに存在している。Ｄｉｖｘパック挿入後のビデオグループとＤｉｖｘパックのレイアウトを図１８に示す。グループ２のＤｉｖｘパックは、同一パックのグループ１のビデオが存在していても、グループ２の符号化ビデオの第１バイトを含むパックに先行している。

【０１１９】

Ｄｉｖｘパックで創られたＤＶＤディスク画像はＭＥＩ照合を通る必要がある。Ｄｉｖｘパックは有効個人＿ストリーム＿１パックである。これは、パックが有効ＳＣＲを含み、ＤｉｖｘパケットがＤＶＤビデオ仕様書のＶ１５－２９、及び３０において定められた必要な値を含んでいることを意味する。ＰＥＳ＿Ｓｃｒａｍｂｌｅ＿Ｃｏｎｔｒｏｌフィールドは“スクランブルなし”に設定する。Ｏｒｉｇｉｎａｌ＿Ｏｒ＿Ｃｏｐｙフィールドは、“オリジナル”に設定する。各Ｄｉｖｘパックは有効ＰＴＳを含む必要がある。このＰＴＳ値はビデオのＰＴＳ値と関係している必要はない。仕様書のＶ１５－３０頁のＮｏｔｅ２で必要な

ＶＯＢの第１パックを除いては、ＰＴＳデータが続くＰＥＳ個人データはない。

【０１２０】

Ｄｉｖｘパックがその付随ビデオパックに物理的に隣接している必要はない。ナビゲーション、オーディオ、又は他のサブ画像パックはＤｉｖｘパックとその付随ビデオパックの間で起こる。ＤｉｖｘビデオグループがＢｌｏｃｋ、ＰＧＣ、ＶＯＢ、セル、またはＶＯＢＵを完全に含んでいる必要はない。Ｄｉｖｘビデオグループは、各境界に広がることができる。Ｄｉｖｘビデオグループは新たなＢＯＶの始まりで再スタートすることができる。

【０１２１】

タイトル作成者は、Ｄｉｖｘパックの作成を可能、不可能の間で選択できる。この作成はデフォルトで不可能となる。Ｄｉｖｘパックの作成が可能な場合、タイトル作成者は、Ｎｄｉｖｘ計算のために映画の分単位で時間を特定できる。タイトル作成者が書き入れた映画の時間が１５分未満、又は２４０分以上の場合、確認メッセージが現れ、タイトル作成者は書き入れた値を受け入れたり、拒絶したりできる。タイトル作成者は、書き入れた映画時間から決定する計算したＮｄｉｖｘを上書きすることができる。ＭｏｖｉｅＬｅｎｇｔｈの値はＮｄｉｖｘの新しい値では更新されない。Ｎｄｉｖｘの値は２未満、又は２０３０－１以上としては特定できない。Ｎｄｉｖｘの上書き値が有効でなく、変更が失敗の場合、エラーメッセージが現れる。Ｄｉｖｘパックの作成ができる場合は、タイトル作成者は映画時間、又はＮｄｉｖｘのいずれかを特定しなければならない。

【０１２２】

５．　メッセージホールの生成及び再配置

ＲＭシステムにおいて、ビデオを構成するＭＢのいくつかがＭＨとされる。ＭＨとなるＭＢの選択方法が必要である。この選択はＲＭシステムにとって決定的ではない。情報（メッセージ）は、ＭＨとしてのＭＢのコンフィギュレーションを仮想的に用いることで表現されかつ再生され得る。したがって、ＭＨの選択のための可能性のある多数の技術を採用し得る。

【０１２３】

好ましい実施形態において、ＭＨの選択は２つのフェーズからなる。すなわち

、１）ＭＨ候補の初期選択（メッセージホールの生成）に続き、２）これらの候補からの最終選択（メッセージホールの再配置）である。この２フェーズ手法は、いくつかの理由のうち、各Ｂフレームに程良い数のＭＨを有することが望ましいとの理由から採用される。メッセージホール生成（第１フェーズ）では、本来必要な数より多くの候補が選択される。再配置（第２フェーズ）では、１つのＢフレームあたりの最終的なＭＨの数が相当に程良い数になるように、Ｂフレーム中の過剰ないくつかの候補が捨てられる。以下の節に述べるように、その他の種々の理由から候補を捨ててもよい。

【０１２４】

ＲＭシステムの本形態では、ムービー中に配置できるＭＨの総数は、サブピクチャチャネル中の空き領域と、ＭＨの大きさと、ＤＶＤが再生される際のビデオにリアルタイムにＲＭを挿入するためのＤＶＤプレーヤの計算上の容量とにより制限されている。１つのＢフレームあたりのＭＨの平均数は、一般に１２より少ない。ＭＨの配置にあたって考慮すべき点は、（１）ＭＨの位置が予測不能であること、（２）ＭＨが複数フレームに渡ってできるだけ均一に配置されること、（３）ＲＭがＭＨの中に挿入された際にＭＨが視認されにくいことである。ＭＨの位置が予測不能であるべきなのは、例えばＭＨの位置が既知であり又は予測されると、ムービー海賊版製作者が伝達情報（メッセージ）をたやすく改竄することができるからである。ＭＨが複数フレームに渡ってできるだけ均一に配置されるべきであるのは、リアルタイムにＭＨ中にＲＭを挿入するのに必要な計算上の容量が一定でないからである。加えて、単一フレーム又は少数のフレームに多数のＭＨが配置されると、これらのフレームを削除することにより、かなりの量の情報が消失又は除去されてしまうからである。ＲＭ挿入の後にビデオ視聴者にとってＭＨが見えにくい（又は少なくともあまり気付かれない）ことは、ビデオの品質がＲＭシステムによってあまり悪影響を受けないことに通じる。また、ＭＨが見えやすいと、改竄が容易になってしまう。

【０１２５】

１つのアプローチは、人手で又は半自動で各フレーム中の所要数のＭＨを選択することである。例えば、ムービーを構成する各Ｂフレームをコンピュータプロ

グラムが表示し、熟練したオペレータがプログラムを使ってMHとして使われるべきMBを選択するのである。このアプローチは、MHの選択に人間の目と人間の洞察力を用いる点で優れている。例えば、気付かれにくいようにMHが配置されるからである。このようなアプローチは、処理すべきフレームが多数にのぼるので、出費がかさみ、時間がかかり、かつ退屈である。例えば、１１０分のムービーには１５８４００フレームがあるのである。したがって、自動MH生成が重要になってくる。本発明に係るアプローチによれば、乱数ジェネレータを用いることでHMがランダムに生成される。MHの視認性を低減するための他のアプローチもある。最初に１フレームの全てのMBを平滑と非平滑（テクスチャあり）とに分類し、それから「テクスチャあり」の集合の中からMHとしてMBをランダムに選択するのである。これら２つのアプローチは、５．１節及び５．２節においてそれぞれ述べる。

【０１２６】

５．１　MHのランダム生成

MH生成に要求されるのは、その位置が予測可能なパターンに従わないようにすることである。この要求は、MHをランダムに配置することによって満たされる。例えば、疑似乱数ジェネレータを用いてMH配置を取り上げるのである。この方法によれば、１フレームを構成するMBの各ロウに固定数の候補MHがランダムに配置される。例えば、仮に１ロウあたりの候補MHの所要数が３であるものとすると、アルゴリズムにより各MBロウに３個のMHが配置されるが、MHの水平位置は、均一分布を持つ疑似乱数ジェネレータを用いてランダムに割り当てられる。各MBロウについては、既に候補MHとして選択されたMBのリストをアルゴリズムが持つことで、同じMBが重ねて選択されず、かつ全てのロウが候補MHの目標数（例えば３）を持つようになされる。例えば、疑似乱数ジェネレータがたまたま同じMBを２度目に選択したなら、このMBが候補MHのリストの中にあることが見い出され、リスト中に存在しないMBが見つかるまで疑似ランダム選択のプロセスが繰り返される。一旦１フレーム中の全てのMHが選択されると、これらがMBアドレス（ＭＢＡ）の昇順にソートされる。昇順ソートの理由は、RMシステム中の他のアルゴリズムがMHをこの順に処理することに

なっているからである。

【０１２７】

このアルゴリズムの疑似コード表現が、図１９に示されている。このアルゴリズムのパラメータは、Ｗｉｄｔｈ、Ｈｅｉｇｈｔ、ｎＦｒａｍｅｓ、ｎｕｍＭＨｒｏｗ及びｓｅｅｄである。Ｗｉｄｔｈ及びＨｅｉｇｈｔの値は、フレームサイズをピクセル単位で指定する。１セットのフレーム中の各ＭＢロウのために生成された候補ＭＨの数がｎｕｍＭＨｒｏｗである。ｎＦｒａｍｅｓパラメータは、処理されるべきフレームの数を指定する。候補ＭＨのＭＢアドレスリスト、すなわちＭＨＡｌｉｓｔが本アルゴリズムの出力である。

【０１２８】

図１９のｇＭＨｒｏｗのアルゴリズムによれば、Ｗｉｄｔｈ、Ｈｅｉｇｈｔ、ｎＦｒａｍｅｓ、ｎｕｍＭＨｒｏｗ及びｓｅｅｄの値が得られる。そして、ＭＢロウ中のＭＢの数、すなわちＭＢ＿Ｗｉｄｔｈが計算され、ｎｕｍＭＨｒｏｗが０からＭＢ＿Ｗｉｄｔｈまでの範囲に制限される。ここに、ＭＢ＿Ｗｉｄｔｈは１ＭＢロウ中のＭＢの数である。疑似乱数ジェネレータはｓｅｅｄパラメータにより指定された「種」の値によって初期化される。疑似乱数ジェネレータを用いて各フレームの各ロウにＭＨが生成され、ｌｏｗＭＢＡとｈｉｇｈＭＢＡとの間の乱整数に戻る。本アルゴリズムによれば、１フレーム内でＭＨの繰り返しがないことが保証される。そして、ＭＨＡｌｉｓｔの中の全てのＭＢＡが昇順にソートされて出力される。

【０１２９】

５．２　領域分類を伴うメッセージホール生成

更に洗練された他のＭＨ選択のやり方は、人がビデオの歪を許容し得るビデオ中の領域を画定し、これらの領域の中にＭＨを配置することである。このやり方は、空間的、時間的、輝度及び／又は色の特性に従って、ピクチャフレームを異なる領域に分割するものである。ここでは、ＭＢが平滑と非平滑（テクスチャあり）とに分類される。「テクスチャあり」と分類されたＭＢが候補ＭＨとしてランダムに選択される。これら２つのプロセスについては、次の段落において説明する。人間の目は非平滑領域中に比べて平滑領域中の歪みに対して敏感なので、

平滑領域へはMHを配置しない。

【０１３０】

Chun et al.,"An Adaptive Perceptual Quantization Algorithm for Video coding", IEEE Transactions Consumer Electronics, 39(3): 555-8, Aug. 1993と同様のアプローチにより、ＭＢのルミナンスブロックのＤＣＴ係数を用いて、２ステップで分類が実行される。この分類アルゴリズムによれば、ピクチャフレーム中のＭＢがランダムに選択される。取り上げられたＭＢが次の段落で説明するいくつかの基準を満たすならば、当該ＭＢが平滑又は非平滑（テクスチャあり）に分類される。ＭＢが「テクスチャあり」に分類されると、当該ＭＢが候補ＭＨとみなされる。このプロセスは、所要数のＭＢが選択されるまで、又はフレーム中に「テクスチャあり」のブロックが残らなくなる（すなわち、全ての「テクスチャあり」のＭＢが候補ＭＨとして受け入れられる）まで繰り返される。

【０１３１】

分類の前にＭＢが満たさねばならない基準の１つは、それがフレームの境界に位置しないことである。ここで境界とは、ＭＢの最初のロウから最後のロウまで、及びＭＢの最初のカラムから最後のカラムまでからなる、ＭＢの外側の矩形のことである。これらのＭＢを外して考える理由は２つある。第１の理由は、家庭用ＶＣＲのような安価なアナログ記録装置では、これらのＭＢがしばしば特に低い忠実度で再生されるからである。したがって、それらの装置で記録された場合には、これらの領域からＲＭ情報を引き出すことは困難である。第２の理由は、多くのテレビジョン受像機ではこれらの領域が表示されず、ピクチャフレームの表示領域から外にはみ出してしまうからである。これらのＭＢは、例えば黒領域に置き換えられる。この置き換えにより、ビデオの目視性に悪影響を及ぼさずに、これらの領域中のＲＭ情報を効率的に除去することができる。

【０１３２】

ＭＢが満たさねばならない他の基準は、大多数の画素ルミナンス（グレースケール）の値が、受け入れ得る範囲の値であるということである。ＭＢをデコード（伸張）すれば２５６（１６×１６）個の画素値が得られる。例えば、２５％より少なければ画素ルミナンス値は２５から２３０までの範囲外であってよいと定

めることができる。この場合には、画素値のうちの６４個以上が２５より小さいか又は２３０より大きいならば、当該ＭＢは分類の対象とならず、したがって候補ＭＢとはなり得ない。画素値のパーセンテージと、受け入れ得る画素ルミナンス値の範囲とは、アルゴリズムのパラメータである。ただし、必要に応じて調整されるものであり、例示の値（２５％、２５、２３０）に限定されない。本例で採用した特定の値は、実験及び技術者の判断により決定されたものである。この基準のきっかけは、明るすぎたり暗すぎたりする画素が、しばしばＭＰＥＧ再エンコーディング（後述）により、又は例えば家庭用ＶＣＲによるアナログ記録によりクリップ（それぞれ増加又は減少）されることにある。ただし、画素値の範囲が減縮されるので、このクリッピング効果がＲＭ情報の再生又は抽出の能力に悪影響を及ぼすことがある。

【０１３３】

ＭＢが満たさねばならない更に他の基準は、前フレーム中の同じ位置のＭＢがＭＨとなるように選択されていないことである。この基準により、ＭＨは隣接フレーム中の同じ位置に配置されない。同じ位置に出現するＲＭは、視認されやすい（気付かれやすい）。

【０１３４】

平滑か非平滑（テクスチャあり）かというＭＢの分類は、まずＭＢを構成する４個のルミナンスブロックを分類することから始まる。これら４個の分類結果は、ＭＢの分類を決定付けるように、評価されかつ結合される。ここで、あるＤＣＴブロックの水平、垂直及び対角線方向のエネルギをそれぞれＥｈ、Ｅｖ及びＥｄで表すこととする。Ｅｈの値は、次のテンプレート中の「ｈ」で示されたＤＣＴ係数の２乗和により計算される。

【０１３５】

．ｄｈｈｈｈ．．
ｄｄｈｈｈｈ．．
ｖｖｄｄ．．．．
ｖｖｄｄ．．．．
ｖｖ．．．．．．

・・・・・・・・
・・・・・・・・
・・・・・・・・

同様に、Ｅｖ及びＥｄの値は、それぞれ「ｖ」及び「ｄ」のマークが付けられた要素の２乗和により計算される。このテンプレートの各位置は、ルミナンスブロックを構成するＤＣＴ係数の８×８行列の要素を表している。このテンプレート中の左上のＤＣＴ係数は、ＤＣ項である。本ブロックの水平、垂直及び対角線方向のエネルギの平均をＥａとする。これは、Ｅｈ、Ｅｄ及びＥｖの平均である。そのうえで、ブロックが次の規則に従って分類される。

【０１３６】

もし（Ｅａ＜Ｔ１）ならば、

当該ブロックは平滑ブロックである：

さもなければ、

当該ブロックは「テクスチャあり」のブロックである。

【０１３７】

Ｅａは、ブロックが平滑か非平滑かの分類に用いられる。しきい値Ｔ１は、一連の実験により、及び技術者の判断により決定し得る。その値は固定である。しかしながら、このしきい値（Ｔ１）は、ビデオ又はその一部の特性に基づいて自動的に決定することも可能である。そして、その値は時間の経過とともに変動してもよい。例えば、１フレーム毎に異なるＴ１の値を用いる。

【０１３８】

ＭＢを構成する４個のルミナンスブロックの分類から、次の規則に従って当該ＭＢが分類される。

【０１３９】

もし（当該ＭＢ中の３又は４ブロックが「テクスチャあり」）ならば、

当該ＭＢは「テクスチャあり」のＭＢである：

さもなければ、

当該ＭＢは平滑ＭＢである。

【０１４０】

上記分類ルーチンの疑似コードを図２０に示す。この分類ルーチンによってコールされる初期化関数の疑似コードを図２１に示す。はじめに、全てのＭＢが分類に利用し得るものと仮定する。そして、前Ｂフレームで選択されたＭＨの位置が、現フレームでは利用不能としてマークされる。これにより、隣接Ｂフレームの同じ位置にＭＨが出現することを防止できる。ＭＨ選択の他の制約は、ＭＨがフレームの境界にあってはならないということである。これらのＭＢも利用不能としてマークされる。分類ルーチンは、フレーム中の予め特定された部分のみを処理する。つまり、調べられるＭＢの数は最大限が規定されている。サーチされるべきＢフレームのパーセンテージを制限することにより、候補ＭＨとして取り上げられるＭＨのパーセンテージが低減される。したがって、隣接フレームの同じ位置に２個のＭＨが出現する確率が低減される。なぜなら、サーチされる領域はランダムに決定され、かつフレーム毎に異なるという傾向があるからである。所要数のＭＨを見い出せなかった場合には、アルゴリズムが終了する。

【０１４１】

初期化ルーチンがコールされた後、与えられたＢフレーム中のＭＨのサーチが開始する。アルゴリズムは、フレーム中のＭＨがなくなるか、又は所要数のＭＨが見い出された場合に終了する。ＭＢのロウ及びカラムは、ランダムに選択されたＭＢＡから計算される。ＭＢは、たいていのルミナンス画素値が規定範囲内でないなら、選択されることはない。そして、ＭＢが分類される。ＭＢ分類関数（Ｃｌａｓｓｉｆｙ＿ＭａｃｒｏＢｌｏｃｋ）の疑似コードを次に示す。ＭＢが「テクスチャあり」に分類された場合には、当該ＭＢが取り上げられるようにマークされる。ロウ上のＭＨの数が１ロウあたりの最大許容数に等しいならば、当該ロウからはもはやＭＨを取り上げない。この制約は、再エンコーディングにとって必要である。ＭＢが平滑であると分類された場合には、これが用いられるこてはない。

【０１４２】

図２２にはＭＢ分類の疑似コードが示されている。８×８ブロックのＤＣＴが計算された後、ＤＣＴ行列中の高エネルギ成分が水平、垂直及び対角線方向のエネルギＥｖ、Ｅｈ及びＥｄとして計算される。これら３値の平均、すなわちＥａ

が計算される。「テクスチャあり」との分類の８×８ブロックの数が数えられる。もし３個又は４個のブロックが「テクスチャあり」ならば当該ＭＢは「テクスチャあり」に分類され、さもなければ分類は平滑となる。

【０１４３】

５．３　メッセージホールの再配置

ＭＨ選択の第２の（かつ最終の）フェーズは、再配置である。再配置の目的は、フレーム間でＭＨをできるだけ均一に配置することにある。いくつかのフレームの中にＭＨの塊ができることがないように、各フレーム中に程良い数のＭＨを配置することが好ましい。この配置は、ＤＶＤプレーヤによるリアルタイムＲＭ挿入の間のほぼ一定の計算上の負担となる。また、いくつかのフレーム中に多数のＭＨが配置されると、これらのフレームの削除によって、かなりの量の情報が除去される結果となる。再配置では、過剰の候補を有するＢフレームからいくつかの候補ＭＨを捨てることにより、１つのＢフレームあたりのＭＨの数がほぼ一定に保たれる。

【０１４４】

再配置アルゴリズムは、ビデオを隣接グループ又は隣接セグメントの単位で処理する。これをセクションと呼んでいる。ビデオのグルーピングに対応したセクションは、そのままＤＶＤの中に格納される。セクションは、ビデオのほとんど任意のポイントで開始及び終了することができる。この状況が図２３に示されている。この例では、セクションがあるフレーム中のある特定のＭＢ（左側の小さい黒正方形）から始まっている。そして、当該フレームの残りに続き、いくつかのフレーム全部と、他のフレームの一部（グレー表示）とを経て、最終フレーム中のあるＭＢ（右側の小さい黒正方形）で終わっている。セクションは、任意のタイプのフレーム（Ｉ、Ｐ又はＢ）で開始したり終了したりすることができる。

【０１４５】

本例では、ＤＶＤ上の制限された容量の補助記憶が、ＲＭシステムに割り当てられている。この制限は、１セクションあたり２Ｋバイトである。この制限のために、最小の補助記憶容量を占めるＭＨの方が、大きい補助記憶容量を占めるＭＨより好ましい。この意味から、ＭＨのサイズは、ＭＨに関連する補助記憶のサ

イズ（ビット単位）であると考えられる。再配置は、大きいサイズのMHより前に小さいサイズのMHを処理する傾向がある。

【０１４６】

再配置の基本コンセプトは複雑なものではない。セクション中の各Ｂフレームに対して、そのフレームについては最終の（捨てられない）ＭＨとなるように、１組の候補ＭＨの中から最小のＭＨを選択する。候補ＭＨからこれら最終ＭＨを、再び選択されることがないように除去する。補助記憶の容量（例えば２Ｋバイト）がなくなるまで、又は候補ＭＨがなくなるまで、この処理を続ける。補助記憶がなくなってしまった場合には、残りの候補ＭＨは捨てられる（用いられない）。この処理は、少なくともあるセクションにおいて、１つのＢフレームあたりの程良い数の最終ＭＨを与える傾向がある。もちろん、あるＢフレームについて候補ＭＨが少ししかない場合には、例えば当該フレームが基本的に平滑ＭＢからなる場合がそうであって、Ｂフレームが候補ＭＨを全く持たないこともあり得るので、このようなケースでは最終ＭＨを少ししか又は全く持たない。

【０１４７】

実用上は、上記基本コンセプトにいくつかの改良が必要である。１つの改良は、補助記憶がなくなった場合のＭＨの配置に関するものである。ここで、セクションに５つのＢフレーム（「ａ」から「ｅ」まで）があるものとし、第３番目のＢフレーム（「ｃ」）の後に補助記憶がなくなったものとする。各Ｂフレーム中のＭＨの数は、

Ｂフレーム	ａ	ｂ	ｃ	ｄ	ｅ
ＭＨの数	７	７	７	６	６

のようになる。

【０１４８】

最初の３個のＢフレームの中には７個のＭＨがあるが、最後の２個のＢフレームの中には６個のＭＨしかない。好ましい配置は、

Ｂフレーム	ａ	ｂ	ｃ	ｄ	ｅ
ＭＨの数	７	６	７	６	７

である。

【０１４９】

（「ｃ」の後に）補助記憶がなくなったとき、Ｂフレーム「ａ」が最後に処理された状態までアルゴリズムが戻る。この時点では、各Ｂフレームに６個のＭＨが割り当てられる。候補ＭＨは、好ましい配置が得られるように、フレーム「ａ」、「ｃ」及び「ｅ」の各々から選択される。あるフレームが利用可能な候補ＭＨをもはや持たないならば、利用可能なＭＨが発見されるまで両方向に動くこととなる。

【０１５０】

基本コンセプトに対するもう一つの改良は、セクションの始まりと終わりに発生し得る部分フレームに関するものである（図２３参照）。セクションの最初もしくは最後のフレームがＢフレームの場合、部分Ｂフレームを生成する必要があるかも知れない。この場合、部分Ｂフレーム用に選択されるＭＨ数はセクション内に存在するＢフレームの比率に従って割り当てられる。例えば、Ｂフレーム領域の４０％がセクション内にある場合、部分フレーム用に選択されるＭＨ数は全フレーム用に選択される数の４０％とすべきである。

【０１５１】

アルゴリズムの擬似コードを図２４に示す。全ての変数をｉｎｉｔｉａｌｉｚｅ（）ファンクションで初期化する。ｉｎｉｔｉａｌｉｚｅ（）ファンクションを図２５に示す。セクションの始まりがＩフレームもしくはＰフレームにある場合、ＩフレームまたはＰフレームに続く最初のＢフレームを最初のフレームとして選択し、そのフレームの最初のＭＢをセクションの開始ＭＢとして採用する。同様に、セクションがＩフレームまたはＰフレームで終わっていれば、そのセクションの最後のＢフレームがセクションの最後のフレームになる。

【０１５２】

セクションがＢフレームの中央にある場合、フレームの一部を前のセクションの前の呼出しで処理する。次のセクションのために、この情報を格納して前のセクションで選択されたＭＨが再度選択されないようにする必要がある。ＤＶＤ境界線に沿うＭＨは、再配置に使用されない。

【０１５３】

初期化の後に（図２４参照）、再配置が始まる。最初のｗｈｉｌｅループは、利用可能なMHがなくなるか、又は選択されたMHのサイズが１セクションあたりの最大サイズ（例えば２Kバイト）を超えた場合に終了する。各フレームから探し当てられた最小サイズのMHは、最終MHとして選択される。

【０１５４】

最初の再配置がなされた後、最後の繰り返し中で選択されたMHが再度の再配置にかけられる。最後の繰り返し中で選択されたMHは除去される。更に均一な配置の中から他のMHが選択される。全てのMHが選択された後、これらがアルゴリズムから出力される。

【０１５５】

６．ＭＰＥＧ再符号化

ＭＰＥＧ再符号化は、RMの挿入後、一以上のMHを備えた一列内のＭＢのビット総数を、RM挿入前の総数と等しく維持することを目的とする。MHの余分なビット数は、より少数のビットとすることで、その列内のＭＢに充填される。原符号化と再符号化のビット数をそれぞれその列用にOrigNumBItsとRowBitCntと定義し、その差を次のようなBitsDiffと定義する。

【０１５６】

BitsDiff=OrigNumBits-RowBitCnt　（６．１）

本発明に対応する符号化のためにより少ないビット数を用いるＭＢを作成するアプローチは、要素ｍｑｕａｎｔを尺度化しながら量子化を減少することである。一般に、ＭＢのＭｑｕａｎｔをより高くすればするほど、ビット数の利用はより減少するが、より大きな歪みが生じる。このため、一般には、ＭＢのＭｑｕａｎｔ値の変化が正確にOrigNumBits以下かそれと同等の（つまりBitsDiff<=0）Rowbitsを作成する。ビットの詰め込みは両者をその後等しくするために利用される。計算の複雑さは増すが、ＭＢの幾つかのＤＣＴ係数の歪みをより調整することにより、ビットを詰め込むためのわずかなビットが得られる。

【０１５７】

各ＭＢのＭｑｕａｎｔの増加量は分からないので、反復捜索は再符号化が必要となる。この反復捜索は、BitsDiffがゼロ以下の状態ならば、終了する。そして

繰り返し数はｉｔｅｒＭａｘと定義された許可数に到達するか、もとのｍｑｕａｎｔ値の増加は最大許容歪み値ＭｑｕａｎｔＩｎｃＭａｘを越える。後者の二つのケースでは、その列内の全てのMHは除かれ、ＲＭメッセージビットを送信するために、その列内のＲＢは全く使われない。ＭＢ内で歪みを統一的に拡散するために、ＭＢのｍｑｕａｎｔ値は同等の量だけ増加される。再復号化の過程をスピードアップするために、全ての必要な符号化情報は、一以上のMHを含む列群のためのＭＰＥＧビット列を復号することで抽出される。この情報は動きベクトル、符号化ブロックパターン、原ｍｑｕａｎｔ値、動きとＤＣＴタイプ、ＭＢが飛ばされているかいないかの決定を含む。もし列内にMHが全く存在しないなら、その列の再符号化過程は飛ばされ、原ＭＰＥＧビット列が使われる。

【０１５８】

【表７】

iterStatus	説明
Inc_delta_Row	再符号化後のビット数が元のビット数にまだ程遠い。BitsDiff<=0 の条件を最初に満たす前の反復状態として定義される。
Dec_delta_Row	再符号化後のビット数が元のビット数にまだ程遠い。BitsDiff>=0 の条件を最初に満たす前の反復状態として定義される。
Inc_Mquant_Row	再符号化後のビット数をできるだけ元のビット数に近づけるために状態 Dec_delta_Row に到達後に設定される反復状態。
Dec_Mquant_Row	再符号化後のビット数をできるだけ元のビット数に近づけるために状態 Inc_delta_Row に到達後に設定される反復状態。
Inc_Mquant_MB	再符号化後のビット数が元のビット数に近い。BitsDiff<=0 の条件を二度目に満たす前の反復状態として定義される。
Final_Iteration	BitsDiff=0 の条件を満たした後の反復状態として定義される。mquant 値の変更は不可。もう一度再符号化してＭＨのビデオストリームデータとビットストリームデータを送出する。その後、反復ループを出る。

再符号化の反復状態

【０１５９】

各繰り返しが読み込まれている間、再符号化状態の決定を助けるために、そのプログラムは、表７で定義された多様なｉｔｅｒＳｔａｔｕｓを使用し最新化する。ｉｔｅｒＳｔａｔｕｓであるInc__delt RowとInc__Mquant__Rowは、各選択されたデルタか１により列内の全てのＭＢのｍｑｕａｎｔ値を増加させるために、そのプログラムを指示するよう使われる。同様にして、ｉｔｅｒＳｔａｔｕｓであるDec__delt__RowとDec__Mquant__Rowは、各選択されたデルタか１により列内の全てのＭＢのｍｑｕａｎｔ値を減少させるためにそのプログラムを指示するよう使われる。

【０１６０】

６．１　ＭＰＥＧ再符号化技術

本発明における第１の実施形態の再符号化によると、一のＭＢが選択され、そのｍｑｕａｎｔ値は一つだけ増加し、全体の列は再符号化される。もし上記した最終条件が満たされるなら、繰り返し過程は止まる。さもないと、もう一つのＭＢが選択され繰り返し過程は継続する。ｍｑｕａｎｔ値は各繰り返しに対して維持され、前の繰り返しに残ることはないであろうことに注意しなさい。ＭＰＥＧ符号化にとり、前のＭＢのｍｑｕａｎｔ値と異なるだけの場合、一のＭＢのそれは符号化され送信されので、ある列の選ばれたＭＢは連続して使われる。その列の最後のＭＢが届いた時、最初のＭＢは次の選択のために再び選ばれるだろう。図２６はこの繰り返し過程の典型的な結果を示す。最初の（ｊ＋１）のＭＢのｍｑｕａｎｔ値が（ｄ＋１）の値だけ増加され、ＭＢの最後のその値がｄの値だけ増加され、そこでは、MB__widthはある列内のＭＢの総数となる。

【０１６１】

このアプローチのアルゴリズムが図２７に表わしてある。６行目のであるように、現フレームの各MH列にとり、（一回で一つの）列内で選択されたＭＢのｍｑｕａｎｔ値は引き続き１だけ増加される。各再符号化が行なわれた後、BitsDiffの条件がゼロ以下を満足してないなら、その列の次のＭＢが２１行目であるように選択される。このアプローチの利点は、実行は容易だが、多数の繰り返しが必要となる。

【０１６２】

第２に実施形態では、表７における多様なｉｔｅｒＳｔａｔｕｓはInc__Mquant__Rowとなるように最初にセットされる。各繰り返しのために、その列内の全てのＭＢのＭｑｕａｎｔ値はデルタにより増加され、全ての列は再符号化される。デルタが１であると仮定しよう。このプロセスはBitsDiffがゼロ以下になるまで続く。そして、ｉｔｅｒＳｔａｔｕｓはInc__Mquant__MBにセットされ、そのｍｑｕａｎｔ値は、そのデルタを減じることにより前の繰り返しのInc__Mquant__MBに再セットされる。次に、その列の最初のＭＢから始め、第１のアプローチはBitsDiffがゼロ以下になるまで使われる。ｉｔｅｒＳｔａｔｕｓは、その後、Final__iterationと等しくセットされ、最後の繰り返しは、映像とＭＢのビット列を送信するための繰り返しループを通して行なわれる。

【０１６３】

本発明の側面である第２の実施形態を説明するアルゴリズムが図２８に与えられている。多様なｉｔｅｒＳｔａｔｕｓは、３行目にあるようにInc__Mquant__Rowと等しく最初にセットされる。各読み込みに対して、列内の全てのＭＢのｍｑｕａｎｔ値は、その列を符号化するのに必要なビット数が標的となるビットレート以下になるまで、７行目にあるようにステップサイズのデルタだけその後増加される。しかしながら、再符号化された列のサイズを可能な限り元の列のサイズに近づけるために、その列内のＭＢの全ての本ｍｑｕａｎｔ値は、２５行目にあるようにデルタにより減少され、そして、ｉｔｅｒＳｔａｔｕｓは２６行目にあるようにInc__Mquant__MBと等しくセットされる。次に、２７行目にあるようにその列内の最初のＭＢから始め、各ＭＢのｍｑｕａｎｔ値は、その列を符号化するのに必要なビット数が標的となるビットレート以下となるまで、最初の実施形態でのように一度につき一つ変化される。この点で、ｉｔｅｒＳｔａｔｕｓは、Final__Iterationと等しくセットされ、最後の読み込みが映像とＭＢのビット列データを送信するための繰り返しループを通して行なわれる。

【０１６４】

第２の実施形態はより多くのプロセスが必要となるが、第１のアプローチと比べた繰り返しの節約はかなりである。Numlter(i)をｉ番目のアプローチのための繰り返し数と定義する。第１の実施形態では、次の式で表わされる。

【０１６５】

Numlter(1)=1+d×MB__width+(j+1)+1

(6.2)

Numlter(2)=1+(d+1)+(j+1)+1

(6.3)

両実施形態においては、最初の繰り返しは、ＭＢの原ｍｑｕａｎｔ値を用いた最初の読み込みのためであり、最後の繰り返しは、映像とＭＢのビット列データを送信するための最後の読み込みのためである。このため、第１のアプローチを越える第２のアプローチを用いた繰り返しの節約は、次の式

Numlter(1)-Numlter(2)=d×(MB__width-1)-1　(6.4)

で与えられる。

【０１６６】

７２０×４８０画素のフレームにとり、一列に４５ＭＢが必要となる。典型的なＭＰＥＧビデオでは、ｄ＝４である。この場合、１７５回の繰り返しを節約できる。

【０１６７】

本発明によるＭＰＥＧ再符号化のための第３の実施形態は、第２の実施形態を越え洗練されたものであり、そこでは図２６におけるｊ値が繰り返しの前後の間で列内の全てのＭＢのビット数の相違を比較することで計算される。その後、列内の最初のＭＢから始める代わりに（図２６のように）列内のＭＢのｍｑｕａｎｔ値を用いることにより、その再符号化は第１の実施形態と切り替わる。ｊ値の概算が一般に現実の値と非常に近いので、BisDiffの状態をゼロ以下に到達させるために繰り返し数を増やす数は、一回か二回と非常に少ない。二番目のアプローチを越える三番目のアプローチの繰り返し数の節約はｊ値に基づくことになる。ｊは０と(MB__width-1)の間のいかなる値をも取りうるので、平均的な節約数は、MB__width/2となる。再び７２０×４８０画素のフレームにとっては、一列内に４５ＭＢが存在する。それは、約２２の繰り返しが節約されることを意味する。

Ｂｉをｋ番目の繰り返しのｉ番目の符号化のビット数と定義する。全てのＭＢ上

のＢｉ（ｋ）の合計はｋ番目の繰り返しのRowBitCnｔと等しい。図２６におけるｍｑｕａｎｔの値を増加するために、（ｄ＋２）の繰り返しの後、BisDiffがゼロ以下の状態になることを、その図は示している。PrevBitCntを前のRowBitCntと定義する、つまり、（ｄ＋１）の繰り返しのRowBitと定義する。その後、ｊは図２９のように決定される。

【０１６８】

第３の実施形態を説明するアルゴリズムが図３０に表わしてある。このアルゴリズムは第２番目のアルゴリズムと似ており、主要な相違は２６～２８行目にあるとうりで、ｊ値前述の記載で計算される。第２の実施形態でのように最初のＭＢの代わりにｊ番目のＭＢからのデルタにより、ＭＢのｍｑｕａｎｔ値は軽減される。その後、ｊ番目のＭＢは、ｍｑｕａｎｔ値を増加させるために、次のＭＢに選ばれる。

【０１６９】

６．２　再符号化時における不適当なメーセージホールの検出と除去

本発明の他の態様では、再符号化処理の際に、メッセージビットの挿入のために用いるべきでない“不適切な”ＭＨも、決定され、およびまたは、除去される。ＭＨは、以下の４つの理由のために、検出され、およびまたは、除去される。

【０１７０】

ＤＶＤセクタ境界：

もし、ある列のメッセージホールが再符号化後にＤＶＤセクタ境界に存在するとすると、そのＭＨは検出され、この情報は第２の処理GenRmgPckに報告される。元のＭＨは、２つに分かれたＭＨとして扱われるが、フラグは、その２つのＭＨが同一のメッセージビットに対応することを示すものとして用いられる。これは、ＤＶＤディスクでは、ＭＰＥＧビデオは、各２ＫバイトのＤＶＤセクタに分割されるからである。このため、ＤＶＤセクタ境界に存在するＭＨは、２つの異なるＤＶＤセクタに分かれて配置されるので、ＲＭ挿入のためのハードウェアのインプリメーションが、さらに複雑になり、コストが高くなる。

【０１７１】

再符号化エラー：

もし、ある列の再符号化の際に、妥当な回数の試行の後に、MHの付加ビットをその列内の他のMBから補償できないことが分かったら、その列にはMHは挿入されない。さもなければ、再符号化の前に存在した問題、例えばMPEG符号化におけるバッファのアンダーフローやオーバーフローが生じるかもしれない。

【0172】

弱いRMキャリア：

もし、あるMHについての、再符号化後におけるRMキャリアの0と1のビデオデータ間の差が極めて小さいときは、このMHは用いられない。この条件の目的は、RMメッセージビットを回復するための、RMキャリアからの妥当な“信号”強度があることを保証することにある。

【0173】

スキップされたMH：

再符号化の後、もしあるMHがスキップされたMHとして決定されたときは、そのMHは除去される。これは、スキップされたMHからは、RMキャリアに付加されたメッセージ情報は回復できないからである。

【0174】

6．3　要約：MH候補、RMキャリアおよび再符号化

図31はすでに述べたような、メッセージホール候補とランニングマーキングキャリアを生成するとともに再符号化を行う処理を要約した図である。これらの処理を行う他の方法を選択するために、表8に、4種類の異なる処理モードが定義されている。

【0175】

【表8】

様々なＭＨ処理モード

ＭＨ処理モード	ファンクション	文字	説明
MHProcMode[0]	ＭＨ生成	0	ＭＨをファイル MHin から読み出す。
		1	ＭＨを分類により無作為に生成する。
MHProcMode[1]	再符号化実行	0	再符号化を完全にスキップする。
		1	再符号化をＭＨ付き行用に実行する。
MHProcMode[2]	ＭＨのビットストリーム書出し	0	ＭＨのビットストリームデータを MHBitOut に書き出さない。
		1	ＭＨの不可ビットストリームを MHBitOut に書き出す。
MHProcMode[3]	ＭＨのビデオデータ書出し	0	ＭＨのビデオデータを MHVideoOut に書き出さない。
		1	ＭＨのビデオデータを MHVideoOut に書き出す。

【０１７６】
上述したように、入力ＭＰＥＧビットストリームＭｐｅｇＩｎがＭＰＥＧ復号

器によって復号され、再符号化に必要な全ての符号化情報が得られる。あるフレームについてMHを生成するために、RMキャリアを生成するために用いられるMBAや他のパラメータが、入力ファイルMHｉｎから読み込まれるか、または上述したようなクラス化によって生成される。このファイルは、MHをランダムに一列一列生成した出力ファイル、再符号化後のMHファイル、またはユーザによって編集されたファイルなどである。もし、MHをクラス化によって生成する手法がとられたときは、MHを生成するのに必要なフレーム全体が再構築され、そのフレームについてクラス化が実行される。いくつかのMBに対してRMキャリアを生成するために、ＤＣＴ係数の代わりにビデオデータを直接変更する手法がとられたときは、これらのMBのビデオデータは再構築される。生成されたMHは、オペレータによって、あるいは再符号化処理を実行することによって、除去される。MHのサイズ、位置および付加ビットストリームは、復号化されるMHのビデオデータとともに、再符号化処理の際に生成され、出力ＦＩＦＯを用いる再配分プログラムに送られる。MHの再配分処理は、MHの平均個数を、要求されるMHレートに可能な限り近づけるために、行われる。キャリア０とRMGパックを含む最終的なＭＰＥＧビットストリームは、第２のプロセスＧｅｎＲｍｇＰｃｋによって生成される。RMGパックは、MHのサイズ、位置オフセットおよびキャリア１ビットストリームを格納する。

【０１７７】

図３２はMH生成に用いられるＧｅｎＲｍプログラムのアルゴリズムを示している。このプログラムは、まず、ライン１に示すように、入力パラメータファイルからMH再符号化モードを読み出す。このプログラムによって、再符号化モードの値に応じて選択される各MH処理モードが、表８に示されている。それから、ライン４に示すように、画像入力ファイルからの各フレームは復号化され、この復号化情報は記憶される。ライン７に示すように、最初の実行時、MH情報を記憶するためにメモリーが割り付けられる。プログラムは、毎回、シーケンスの最後に達したか否かをチェックする。もしそうであれば、ライン１０に示すように、プログラムは再配分処理を呼び出して、残りの処理されたフレームMHを均等に配分する。もし、現フレームがＢフレームであり、MHをクラス化けによっ

て生成する手法がとられたときは、ライン１６に示すように、プログラムはクラス化プログラムを呼び出して、フレームにMHを生成する。次に、ライン１７に示すように、再符号化機能が呼び出されて、MHを含むそのフレーム内の列を再符号化する。再符号化機能は、そのフレームのMHのビデオおよびＭＰＥＧ情報を出力ＦＩＦＯに書き込む。セッション内の全てのフレームが処理されたとき、ライン２０に示すように、ＧｅｎＲｍプログラムは再配分処理を呼び出して、このセッションで処理された全てのフレームにおいてMHを再配分する。

【０１７８】

再符号化機能は、上述したＭＰＥＧ再符号化の第３の実施形態を利用する。再符号化処理の反復回数を削減するために、再符号化プログラムは、また、再符号化の際の反復処理で定義された反復状態Inc__delta__Row を選択することによって、より大きなデルタ（１よりも大きい）の利用を選択することも可能である。再符号化機能は、その列の全てのＭＢのｍｑｕａｎｔ値を増加させるために、反復状態Inc__delta__Row またはInc__Mquant__Rowを、選択値デルタか１かによって、それぞれ選択することができる。また、再符号化処理をスピードアップするために、前の列のｍｑｕａｎｔインクリメントｄを、次の列などののｍｑｕａｎｔインクリメントの初期見積もりとして、用いる。この場合、前の値ｄが大きすぎるときは、再符号化プログラムは、反復処理で定義された反復状態Dec__delta__Row を選択することができる。反復状態Dec__delta__Row およびDec__Mquant__Rowは、その列の全てのＭＢのｍｑｕａｎｔ値を減らすために、選択値デルタか１かによって、それぞれ選択することができる。

【０１７９】

再符号化プログラムはまた、再符号化される現フレームについて、その前のイントラまたはノンイントラ量子化マトリックスが変えられるか否かを検出する。この情報は、その後、現フレームの全MHについてMHキャリアを生成する間に、対応するＤＣＴ係数に付加されるシグマ値（ノイズ）を、スケールするために用いられる。プログラムはチェックし、MHを含まない列の再符号化をスキップする。再符号化される列の反復状態は、最初は、Inc__delta__Row に初期化される。その列のＭＢの元のｍｑｕａｎｔ値は、最初は、前に再符号化された列のｍ

ｑｕａｎｔインクリメントｄによってインクリメントされる。反復ループの各繰り返しの際に、プログラムは、その列のＭＢのｍｑｕａｎｔ値を調整することによって、ＭＨの補償を試行する。ＭＢのｍｑｕａｎｔ値の変更に用いられる正確な手法は、反復ループの各繰り返しの際の反復状態の値に依存する。

【０１８０】

もし、予め設定した最大回数の繰り返し処理の後にも、目標ビットレートが達成できない場合は、その列の全ＭＨは除去され、元の列は置換されない。また、もし、ある列に必要なｍｑｕａｎｔインクリメントｄが、予め設定した許容最大歪み値よりも大きくなったときは、その列の全てのＭＨが除去される。ＭＨに対応するＭＢに対して、プログラムは、再符号化ループの最初の繰り返しの際に一度だけ、ＭＨブロックをＲＭとともに、ＲＭなしで生成し記憶する。ＭＨに対応するＭＢは、それから、特定のメッセージホール置換数（例えば３：オリジナル、キャリア０およびキャリア１）と等しい回数だけ、再符号化される。このそれぞれ再符号化において、対応する符号化ビットストリームは、異なる列バッファに、ＭＨ置換子のバイト列とともに記憶される。

【０１８１】

各ＭＨのキャリア０および１についてのＭＰＥＧビットストリームデータは、プログラムにおいて、入力パラメータファイルからの各キャリアのそれぞれの値を読むことによって、選択（制御）される。これらの値は、キャリア０および１についてのＭＨ置換指標値である。このＭＨ置換指標値によって、キャリアのビットストリームデータは、この値に一致する列バッファから選択される。もし、ＭＨのキャリア１と０のビデオデータ間の差が零であるときは、プログラムはこのＭＨを無効としてマークする。キャリア０として選択された（列バッファの１つから）ビットストリームデータは、常に、各ＭＨに対応するＭＢ内に挿入される。このように、ＧｅｎＲｍプログラムによって生成された出力ＶＯＢは、各ＭＨについてキャリア０ビットストリームデータを含む。

【０１８２】

再符号化処理をスピードアップするために、その列のＭＰＥＧビットストリームは、最後の繰り返しの際にのみ、ローバッファ（Row Buffer）に書き込まれる

。それ以外は、プログラムは、各繰り返しにおいて、その列を再符号化するために必要なビット数を計数するのみである。また、その列のＭＢは、最後の繰り返しの際に再び量子化されない、というのはｍｑｕａｎｔ値は変わらないから。その列のＭＰＥＧにおいて、元のサイズ、オフセット、ＭＰＥＧビットストリームおよびビデオデータは、各MHのキャリア０および１とともに、最後の繰り返しの際にのみ、出力ＦＩＦＯに書き込まれる。各MHのキャリア１のビットストリームデータは、ＭＨＰｒｏｃＭｏｄｅ［２］の値が“１”のときにのみ、第２の処理ＧｅｎＲｍｇＰｃｋによってファイルＭＨＢｉｔＯｕｔに書き込まれる。このファイルは、この後に、メッセージビット変調に用いられる。もし、ＭＨＰｒｏｃＭｏｄｅ［３］の値が“１”のときは、第２の処理ＧｅｎＲｍｇＰｃｋがまた、その元の値を、各MHのキャリア０および１の両方のビデオデータとともにＭＨＶｉｄｅｏＯｕｔに書きこむ。ＭＨＶｉｄｅｏＯｕｔからの情報は、その後に、元のビデオの各MHに挿入された各メッセージビットを回復するために用いられる。プログラムは、その終わりに、ビットスタッフィングを実行し、その列の元のサイズと同一サイズを確保する。最後に、プログラムは、再分配を行う現フレームの全ての更新されたMHについて、MH情報（seed、ＭＨＡなど）を記憶する。

【０１８３】

本発明の他の態様によると、再符号化処理をさらに高速化するために、ランニングマークのＭＰＥＧ２再符号化を、他のＣＰＵおよびまたは他の装置によって、並行して実行してもかまわない。生成された全てのＲＭを集めて、再生ユニットの動作をシミュレートするための後処理が必要となる。このように、本発明によると、プログラムＧｅｎＲｍｐＰｃｋが、ランニングマークを生成する２段階処理の第２の処理として用いられる。ＲＭが再符号化ＭＰＥＧ２データの形式で生成された後は、プログラムＧｅｎＲｍｐＰｃｋは、そのＲＭをＤＶＤ仕様のＶＯＢに挿入し、ランニングマークＶＯＢを生成する。ランニングマークのキャリア０はビデオに挿入され、キャリア１はＲＭＧ＿ＰＣＫ セクタに挿入される。この第２の処理は、また、再生ユニットのスループットがリアルタイムでＲＭを挿入するのに十分であることをチェックする。また、ＲＭを含む新たなＶＯＢが

生成されたとき、プログラムＧｅｎＲｍｇＰｃｋによって、挿入されたＲＭの個数を映画の長さに比して評価するチェックがなされる。メッセージのコピーが、プレーヤから送られたメッセージが確実に回復できるように、いくつか補完されるかもしれない。

【０１８４】

プログラムＧｅｎＲｍｇＰｃｋは、実行時にアクティブとなる２個の入力モジュールを有する。第１のモジュールは、入力ＶＯＢから、セクタのストリームを抽出し特定する。第２のモジュールは、ＧｅｎＲｍによって生成されたデータパケットを様々なファイルに読み出して、ＲＭがＭＰＥＧでどのように符号化されるかを理解する。メインモジュールは、ランニングマークキャリア０および１を、ビデオパケットとＲＭパケット（ＲＭＧ＿ＰＣＫ）に、それぞれ挿入する。その出力は、ＧｅｎＲｍから受けたデータかれ生成された回復ファイルとＶＯＢからなり、ＲＭデータをサポートするために調整された後、一度に１セクタずつ書き込まれる。ＭＰＥＧ２ランニングマークを準備するためにＧｅｎＲｍの複数のコピーを用いるとき、プログラムＧｅｎＲｍｇＰｃｋは、ＶＯＢを完全に処理するために、全てのセグメントを集めてまとめる。もし、ただ１つの処理がＲＭを生成したとすると、ｍｈｄａｔａと名付けられた単一ファイルに全てのデータが格納され、ＧｅｎＲｍｇＰｃｋは必要となる全ての情報をこのファイルから得る。もし、１個よりも多いｍｈｄａｔａファイルが生成されたとすると、ＧｅｎＲｍｇＰｃｋはＲＭを、ファイルｍｈｄａｔａ．０００、ｍｈｄａｔａ．００１、ｍｈｄａｔａ．００２など最終ファイル（常にｍｈｄａｔａ．９９９と名付けられる）までからロードする。プログラムはｍｈｄａｔａファイルを読み、最初のＲＭデータを探す。そして、一致するセクタか空きＲＭＧ＿ＰＣＫを見つけるまで、セクタを出力にコピーする。セクター致があると、ＭＰＥＧ２データは置換され、キャリアがＲＭＧ＿ＰＣＫに付加され（ＲＭＧ＿ＤＡＴＡエントリが付加され）、新たなＲＭがｍｈｄａｔａファイルから読み出される。ＲＭＧ＿ＰＣＫがあると、メモリに保持されＲＭＧ＿ＤＡＴＡフィールドに順次詰められた前のＲＭＧ＿ＰＣＫは、出力ファイルにおいてシーク（Seek()）を実行することによって更新され、現ＲＭＧ＿ＰＣＫは次のセクションのためにクリアされる。

【０１８５】

２つの異なるセクタを覆う１個のＲＭが見つかったとき、ＧｅｎＲｍｇＰｃｋはこのＲＭを２個のＲＭに分け、２個のＲＭＧ＿ＤＡＴＡフィールドを生成する。そのＲＭの２つの部分は両方とも、ＲＭＧ＿ＤＡＴＡのbcincrモードフィールドを用いることによって同一のメッセージビットを示している。セクタが読まれるたびに、タイミングチェックがなされ、再生ユニットのＣＰＵによるシミュレーションが、データオーバーフローを避けるためにセクタレベルで実行される。各セクタは、そのタイプ（ＲＭＧ＿ＰＣＫ、ビデオ、その他）やその状態に基づいて、所定の最小処理時間を必要とする。空きＲＭＧ＿ＰＣＫが必要とする時間は、満たされたＲＭＧ＿ＰＣＫよりも少なく、セクタに挿入された各ＲＡＭは、実行時間中にデータ置換のためにより多くのＣＰＵ時間を必要とする。もし、ＲＭの挿入がプロセッサのオーバーフローを引き起こしたときは、そのＲＭは除去される。本発明によると、ＧｅｎＲｍはまた、元のビデオＭＢをＲＭキャリア０および１とともに送る。ＲＭが落ちたとき、元のビデオＭＢはＭＰＥＧ２ビデオエレメンタリストリームに挿入され、キャリア０および１は無視される。各ＲＭはメッセージビットと関連しているので、この場合、メッセージビットカウンタは増加しない。

【０１８６】

さらに、本発明によると、マルチプロセッサコンピュータの処理を高速化するために、ＭＰＥＧ２ビデオストリームを複数の部分に分割するのが好ましい。この場合、特別な配慮が必要になる。というのは、ＭＰＥＧ２符号化は参照フレーム符号化に基づくものだからである。このため、ビデオを適切に符号化するために、少なくとも２つの参照フレームがロードされなければならない。プログラムＧｅｎＲｍおよびＧｅｎＲｍｇＰｃｋは、ＤＶＤ　ＶＯＢ内では、ナビゲーションセクタ（ＮＶ＿ＰＣＫ）がイントラ符号化フレーム（Ｉ）の直前にあるという事実を利用している。図３４に示すように、原ストリームは複数のセグメントに分けられる。ＩはＩフレームの前のナビゲーションパックを表し、ＰおよびＢは予測フレーム（第２の種類の参照フレーム）および双方向予測フレームを表している。ハッチの入ったセルは、各セグメントの、ＧｅｎＲｍおよびＧｅｎＲｍｇ

ＰｃｋがＲＭを挿入している部分を表している。

【０１８７】

図３４から分かるように、第１のセグメント以外の全てのセグメントでは、第２のＩフレームの前のフレームは、ＲＭを挿入することなく復号化される。なぜなら、これらのフレームはその前のセグメントですでに挿入されているからである。しかしながら、ＲＭが挿入されるとき２つの参照フレームが確実にロードされるように、これらのフレームは復号化されなければならない。

【０１８８】

６．４　多重視野角

本発明の他の態様によると、プログラムＧｅｎＲｍおよびＧｅｎＲｍｇＰｃｋは多重視野角をサポートすることができる。多重角を用いると、再生時に、その映画の部分（セグメント）について、複数の利用可能な角度セグメントから、いずれか１つの視野角（ビデオ）セグメントが再生可能となる。しかしながら、多重視野角をサポートするために、それぞれの角度セグメントのＲＭの個数は、一定に保たなければならない。これは、各角度セグメントには同一のメッセージビットが挿入されなければならないからである。また、ＲＭ（各角度セグメント）は必ずしも同一の位置にある必要はないが、総数は、メッセージ回復が可能なように、同一でなければならない。ＲＭの個数が最小の角度セグメントは、他の角度セグメントのために用いるサイズを与える。ＲＭの個数がより多い角度セグメントについては、メッセージ回復能力を増すために、ＲＭのいずれかを落とすか、重複させる（bcincrモードを用いて）。

【０１８９】

図３３は３種類の可能な角度セグメントを示している。ここでは、元のビデオストリームは、処理のために、セグメントＮ１，Ｎ２およびＮ３に分割されている。各ＲＭＧ＿ＰＣＫが適用される範囲は、ＲＭがセグメント境界と交叉しないように、調整される。このため、様々な角度セグメントに先行する最終パック（ＲＭＧ＿ＰＣＫ）について、全てのＲＭは角度セグメントの前にある（セグメントＮ２の先頭）。同様に、各角度セグメントの最終ＲＭＧ＿ＰＣＫは、セグメントＮ２の最後までにのみＭＨを含む。このため、図に示すように、Ｎ２の各角度

セグメントでは、ＲＭは最初は存在し得ない。というのは、最初の空きＲＭＧ＿ＰＣＫが見つかるまでは、ＲＭを記憶する場所がないからである。同様に、セグメントＮ３の先頭のビデオには、セグメントＮ３に最初のＲＭＧ＿ＰＣＫが見つかるまでは、ＲＭは存在し得ない。最後に、プログラムＧｅｎＲｍｇＰｃｋは全てのビデオセグメントを集めてまとめ、上述したように、ＶＯＢを完全に処理する。

【０１９０】

６．５　ルックアップテーブルを用いた高速量子化

再符号化の一部として、ＭＢの列の各ＭＢのＤＣＴ係数を量子化する。量子化の効果的な処理手法として、ＭＢのＤＣＴ係数の全てのあり得る値について量子化を計算し、その結果をルックアップテーブル（ＬＵＴ）に格納するというものがある。ＭＢの量子化に必要な計算処理は、計算コストがこれよりもはるかに少ないテーブルルックアップ処理に置き換えられる。

【０１９１】

ＭＢを量子化するために、以下の式が用いられる。

【０１９２】

ｙ＝（（３２・ｘ＋ｄ／２）／ｄ）／（２・ｍｑｕａｎｔ）

ここで、ｘは８×８ブロックの各画素のＤＣＴ係数値、ｄは画素の量子化行列係数、そしてｍｑｕａｎｔは、８×８ブロックをどの程度粗くまたは精密に量子化するかを定める量子化スケール係数である。この式は、ＭＢの各８×８ブロックについて、輝度成分および色差成分の両方について計算される。ただし、量子化行列は、輝度値と色差値とで異なる。８×８ブロックの６４個のｘ値のそれぞれについて、ｙ値が計算される。再符号化でこの式が演算される回数は多いので、この式の演算を減らすことによって、実質的な計算量を削減することができる。

【０１９３】

上式のｙ値は、幅広い値をとり得る３個の整数値変数（ｘ，ｄおよびｍｑｕａｎｔ）から求められるため、この式を３次元（３インデックス）ＬＵＴにそのままインプリすると、非現実的なほど大容量の記憶媒体が必要になる。このような記憶容量の要求を軽減するために、上式を２つの部分に分けて、２個のＬＵＴに

インプリすることができる。その２つの式は、次のようになる。

【０１９４】

ｙ１＝（３２・ｘ＋ｄ／２）／ｄ

ｙ２＝ｙ１／（２・ｍｑｕａｎｔ）

ここで、ｙ２は上述のｙと同一である。２次元（２インデックス）ＬＵＴが、この２つの式をそれぞれインプリメントするために用いられる。最初の式に係るＬＵＴすなわちＬＵＴ１は、ｘおよびｄをインデックスとする２次元（行列）テーブルである。第２の式に係るＬＵＴすなわちＬＵＴ２は、ＬＵＴ１から検索した値と、ｍｑｕａｎｔをインデックスとして用いる。ＬＵＴ２はｙ２（そしてｙ）のとり得る値の全てを格納する。

【０１９５】

ＬＵＴ１について考える。ｘの値は－２０４８から２０４７までである。もし、例えばｄが１のときは、ｙ１の値は３２×（－２０４８）から３２×２０４７まで（－６５５３６から６５５０４まで）の範囲となる。ｙ１の範囲が大きく、－６５５３６と６５５０４の間の中間値の多くは生じ得ないので、ｙ１の値をＬＵＴ２のインデックスとして直接利用することは（記憶容量の点で）効率的でない。この理由から、ｙ１の値をＬＵＴ１に格納する代わりに、他の種類の整数値すなわちｙｉｎｄｅｘ１が格納される。このｙｉｎｄｅｘ１は、０からある最大値までの連続した整数値となる。ｙ１の各値について、唯一のｙｉｎｄｅｘ１が存在する。言い換えると、ｙｉｎｄｅｘ１とｙ１とは、１対１の対応関係がある。ｙｉｎｄｅｘ１値は、ＬＵＴ２に適切な（効率的な）インデックスを与える。

【０１９６】

要約すると、量子化式を１組の２次元ＬＵＴにインプリメンテーションすることは、次のように表される。

【０１９７】

ｙｉｎｄｅｘ１＝ＬＵＴ１［ｘ，ｄ］

ｙ２＝ＬＵＴ２［ｙｉｎｄｅｘ，ｍｑｕａｎｔ］

ここで、ＬＵＴ１［ｘ，ｄ］は、ｘおよびｄをインデックスとして用いてＬＵＴ１から検索した値を表す。

【０１９８】

結論

述べられたのは、圧縮されたディジタル信号ストリーム、例えばＭＰＥＧ標準でエンコードされたビデオストリームの、好ましくは目立たないようにするために画像の加工された部分に配置され、また、フレームごとに位置を変えるピクセルブロックまたはマクロブロックの形式の中にマークを含むための変換のための方法論である。マークのロケーションは、オリジナルのビデオ（または他の情報）とともに参照媒体に保存される“メッセージホール”によって定義される。マークは、ＤＶＤやテスト媒体などのテスト下の媒体中のメッセージホールに対応する内容を比較することによりデコードされる。マークは、テスト下の媒体が複製されたソースを示す情報、オリジナル媒体や、複製が行われたプレイバックユニットのシリアルナンバー、および複製の時刻などを表す。この情報は、複製者による検出や改変を防ぐために暗号化され、スクランブルされ、好ましくは、多数のソースの識別能を高め、ノイズがあっても復調を可能にするために展開される。

【０１９９】

ここでは本発明のベストモードが述べられたが、本発明は異なるようにも実践され得る。例えば、本発明は事前に記録された媒体の保護の環境について説明されたが、本発明は同様にインターネットや他のコミュニケーション媒体から得られるデータに対しても適用できる。他の変形としては、本発明は、述べられたようなムービーの複製のソースを特定するだけでなく、ディスクや他のタイプの媒体、またはネットワークから複製されたソフトウェアにも適用できる。さらに、開示は保護されたデータの複製者の追跡の例を用いてなされたが、本発明は複製を防ぐためのメッセージデータを用いるためにも適用できる。例えば、本発明は、保護された素材の判読可能な複製がなされ、または有用なプレイバックが行われるのを防ぐためにランニングマークメッセージが適用されるウォーターマークコピー防止のコンテキストに適用され得る。さらに、ランニングマークメッセージは、望まれるなら、機関によって発行されて、顧客にディスクまたは他の形式のメディアまたはネットワーク空のデータの正当な複製を認可する認可コードと

比較され得る。この認可は、例えばＤＶＤセクタをプログラムしたりまたは、顧客がＤＶＤを１回限りまたは複数回再生し、または所定の数の複製を作製するのを可能にするための他の保護の実現をなすために適用され得る。セクタは、別の方法として、顧客がＤＶＤを種々の目的に用いたり、無制限の使用を可能にしたりするのを防ぐためにプログラムされ得る。

【図面の簡単な説明】

【図１】

本発明によるＭＰＥＧランニングマーク位置の作成を示す高レベル機能ブロック図である。

【図２】

コピーのソースメッセージを行う３つのメッセージホールのある画像を示す。

【図３】

ビデオストリームで実施したメッセージホールを示す線図である。

【図４】

（ａ）および（ｂ）は、本発明によるランニングマークシステム符号器及び復号器を示すブロック線図である。

【図５】

データ記号基準を使用したメッセージビットの記号化を示す線図である。

【図６】

（ａ）および（ｂ）は、鎖状誤差訂正符号化及び複合化の線図である。

【図７】

本発明の実施例で実行する包旋状誤差訂正コーダを示す。

【図８】

本発明の実施例で実行するＣＤＭＡ符号化処理を示す線図である。

【図９】

（ａ）－（ｅ）は、ビデオストリームで実施したコピーのソースメッセージのセキュリティを向上させるためのマルチプル可変長配行を示す。

【図１０】

（ａ）－（ｄ）は、本発明によるビット符号化の変形例を示す

【図１１】
本発明に一形態でのＤＣＴキャリア係数へのノイズ組成を加えることによる元の信号波形の広がりを示す。
【図１２】
本発明におけるＲＭカメラ生成のためのＤＣＴの試みを示す擬似コードである。
【図１３】
ＤＣＴの試みを使用した輝度ブロックのためのピクセルドメインにおけるノイズパターンを示す。
【図１４】
（ａ）－（ｄ）は、ランニングマークデータストリームのレイアウトと構成の線図である。
【図１５】
本発明によるＤＶＤ信号ストリームで行われるビット抽出を示す線図である。
【図１６Ａ】
本発明によるランニングマーク生成の動作を示すアルゴリズムである。
【図１６Ｂ】
本発明によるランニングマーク生成の動作を示すアルゴリズムである。
【図１７】
本発明の一形態によるＤｉｖｘマーク挿入前、挿入後のビデオグループのレイアウトを示す。
【図１８】
本発明の一形態によるＤｉｖｘマーク挿入前、挿入後のビデオグループのレイアウトを示す。
【図１９】
ｇＭＨ行を定義するアルゴリズムである。
【図２０】
本発明に一形態によるマクロブロックの分類のための擬似コードである。
【図２１】

分類機能によって要求される初期化（）のための擬似コードである。

【図２２】

マクロブロック分類の他の実施例のための擬似コードである。

【図２３】

処理する典型的なフレーム区分の線図である。

【図２４】

再分配手順の擬似コードである。

【図２５】

再分配機能によって要求される初期化（）のための擬似コードである。

【図２６】

メッセージホールの予備ビットの補間するための流れ行におけるマクロブロックのためのＭｑｕａｎｔの増加数値を示す。

【図２７】

本発明によるＭＰＥＧ再符号化の第一形態を示すアルゴリズムである。

【図２８】

第２形態を示すアルゴリズムである。

【図２９】

ｋ回（ｋ＝（ｄ＋２））繰り返すためのｊの位置決めのアルゴリズムである。

【図３０】

ＭＰＥＧ再符号化の第３形態を示すアルゴリズムである。

【図３１】

本発明によって行われる機能をまとめた高レベル線図である。

【図３２】

ＧｅｎＲＭプログラムを示すアルゴリズムである。

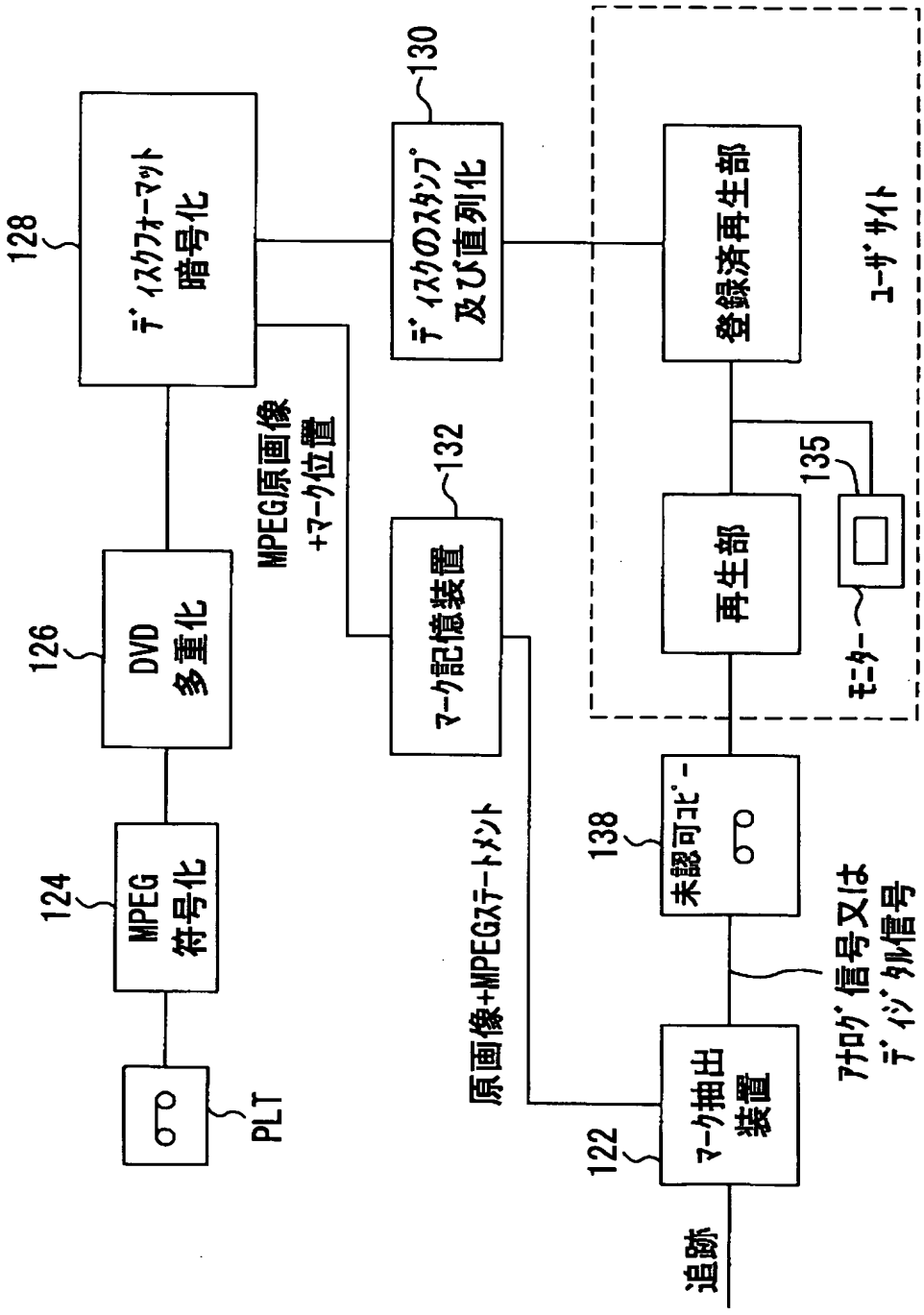
【図３３】

見る角度を変えたビデオ表示を示す。

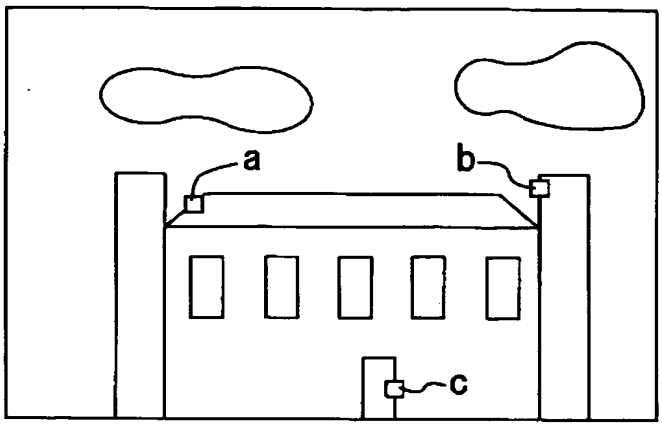
【図３４】

複数のセグメントを示す。

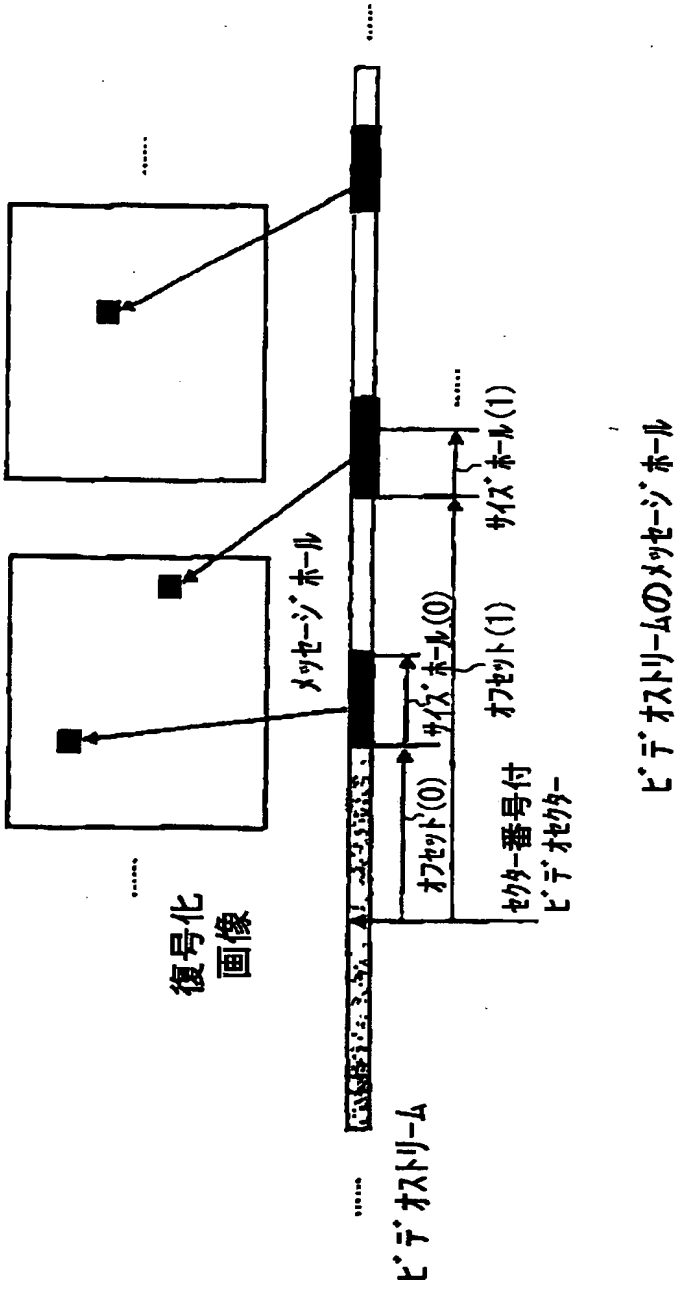
【図１】


128
ディスクフォーマット
暗号化
130
ディスクのスタンプ
及び直列化
登録済再生部
ユーザサイト
MPEG原画像
+マーク位置
132
135
126
DVD
多重化
マーク記憶装置
再生部
モニター
124
MPEG
符号化
原画像+MPEGステートメント
138
未認可コピー
PLT
122
マーク抽出
装置
アナログ信号又は
ディジタル信号
追跡

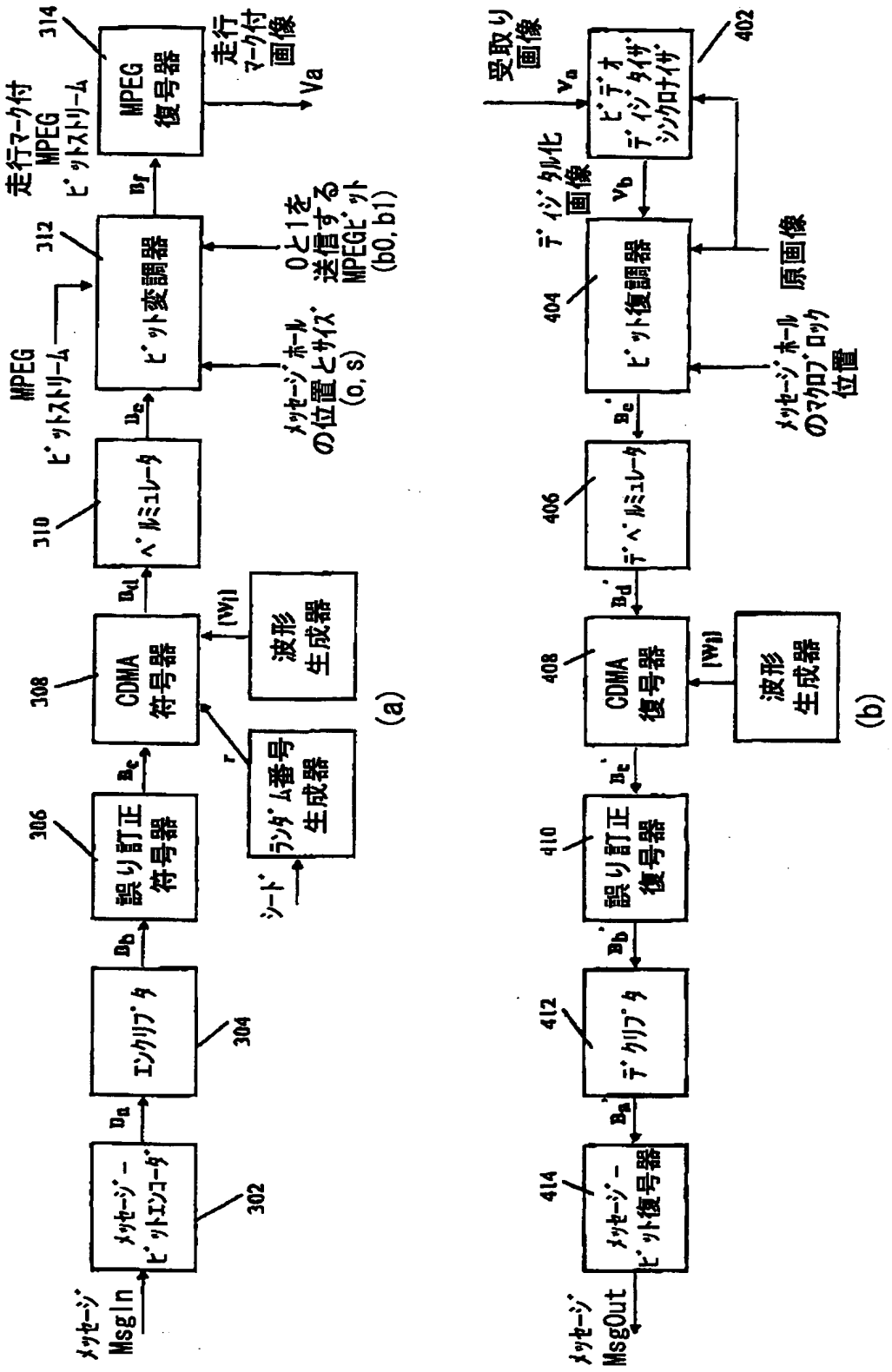
【図２】



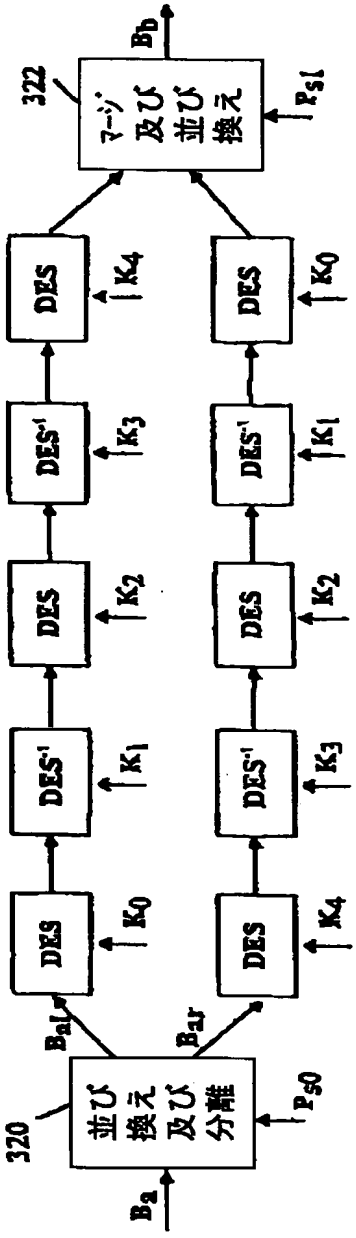
【図３】


復号化
画像
メッセージ ホール
ビデオストリーム
オフセット(0)
サイズ ホール(0)
オフセット(1)
サイズ ホール(1)
セクター番号付
ビデオセクター
ビデオストリームのメッセージ ホール

【図４】

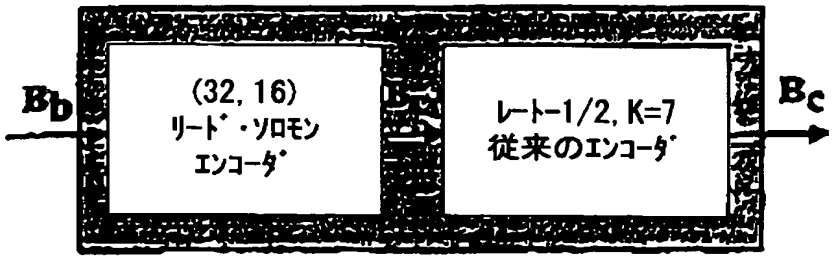

メッセージ
Msg In
メッセージ－ビットエンコーダ
302
エンクリプタ
304
誤り訂正符号器
306
CDMA符号器
308
ベルミュレータ
310
ビット変調器
312
MPEGビットストリーム
MPEG復号器
314
走行マーク付MPEGビットストリーム
走行マーク付画像
Va
0と1を送信するMPEGビット
(b0, b1)
メッセージホールの位置とサイズ
(0, s)
波形生成器
[Wi]
ランダム番号生成器
r
シード
(a)
受取り画像
Va
ビデオデジタイザシンクロナイザ
402
デジタル化画像
Vb
ビット復調器
404
デベルミュレータ
406
CDMA復号器
408
誤り訂正復号器
410
デクリプタ
412
メッセージ－ビット復号器
414
メッセージ
MsgOut
原画像
メッセージホールのマクロブロック位置
波形生成器
[Wi]
(b)

【図５】

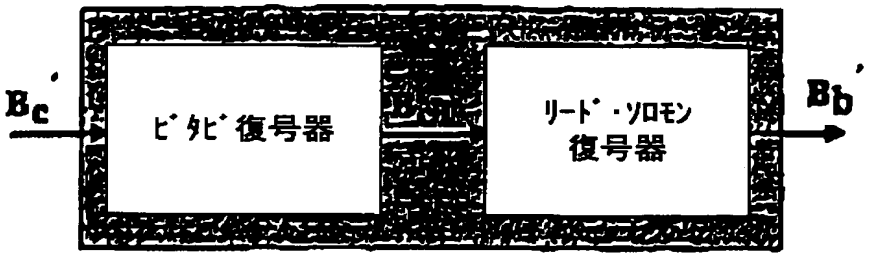


128ビット前並換え及び後並換えでデータ暗号化標準(DES)を用いたメッセージビットの5重暗号化

【図６】

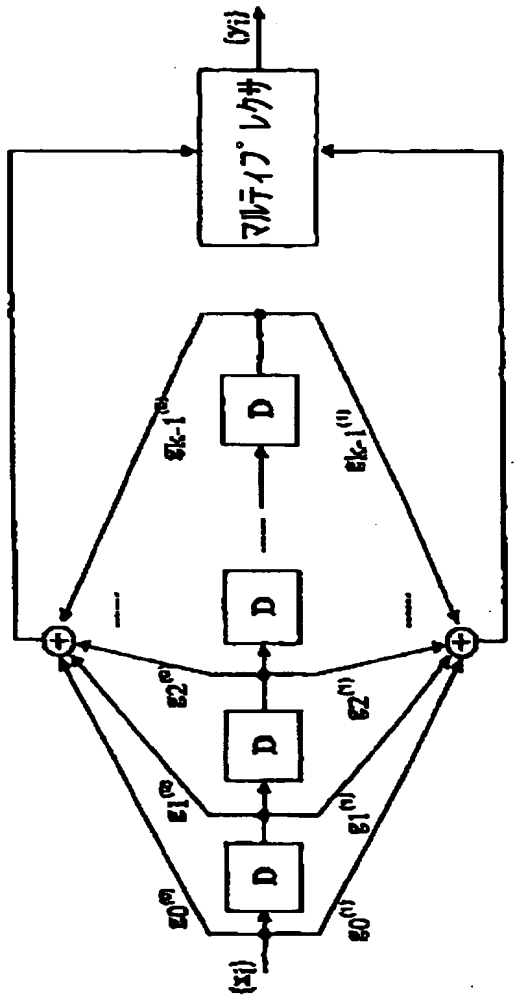

Bb
(32, 16)
リード・ソロモン
エンコーダ
Bc
レート1/2, K=7
従来のエンコーダ

(a)


Bc'
ビタビ復号器
リード・ソロモン
復号器
Bb'

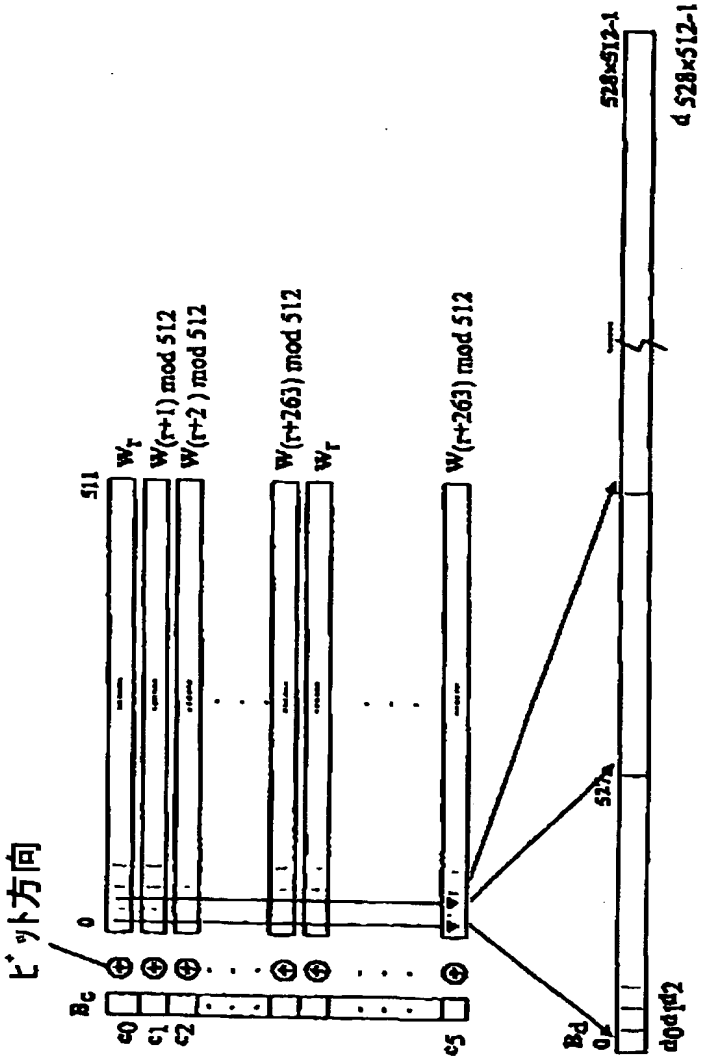
(b)

【図７】



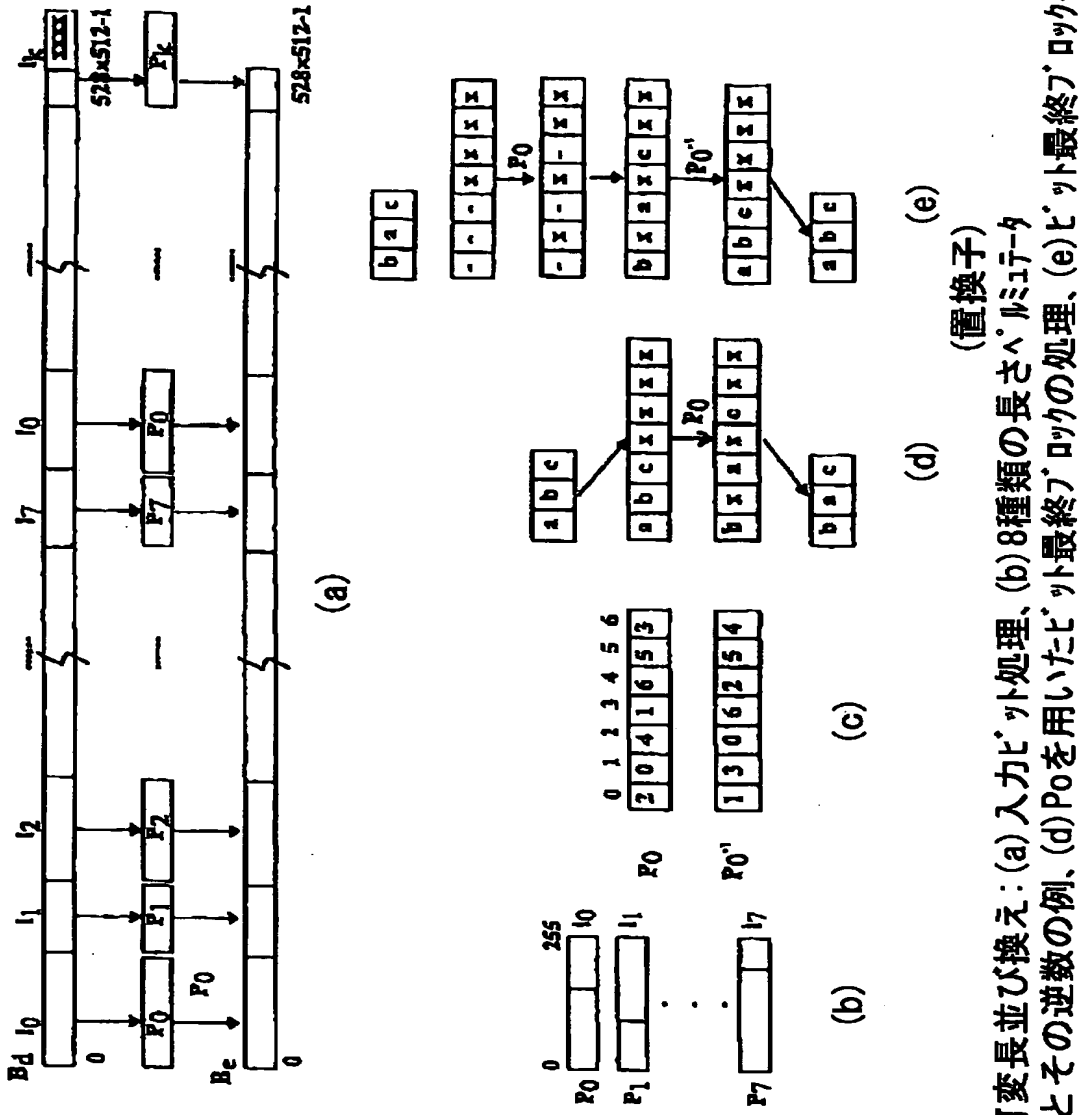
拘束長Kの従来の1/2レート誤り訂正符号器

【図８】



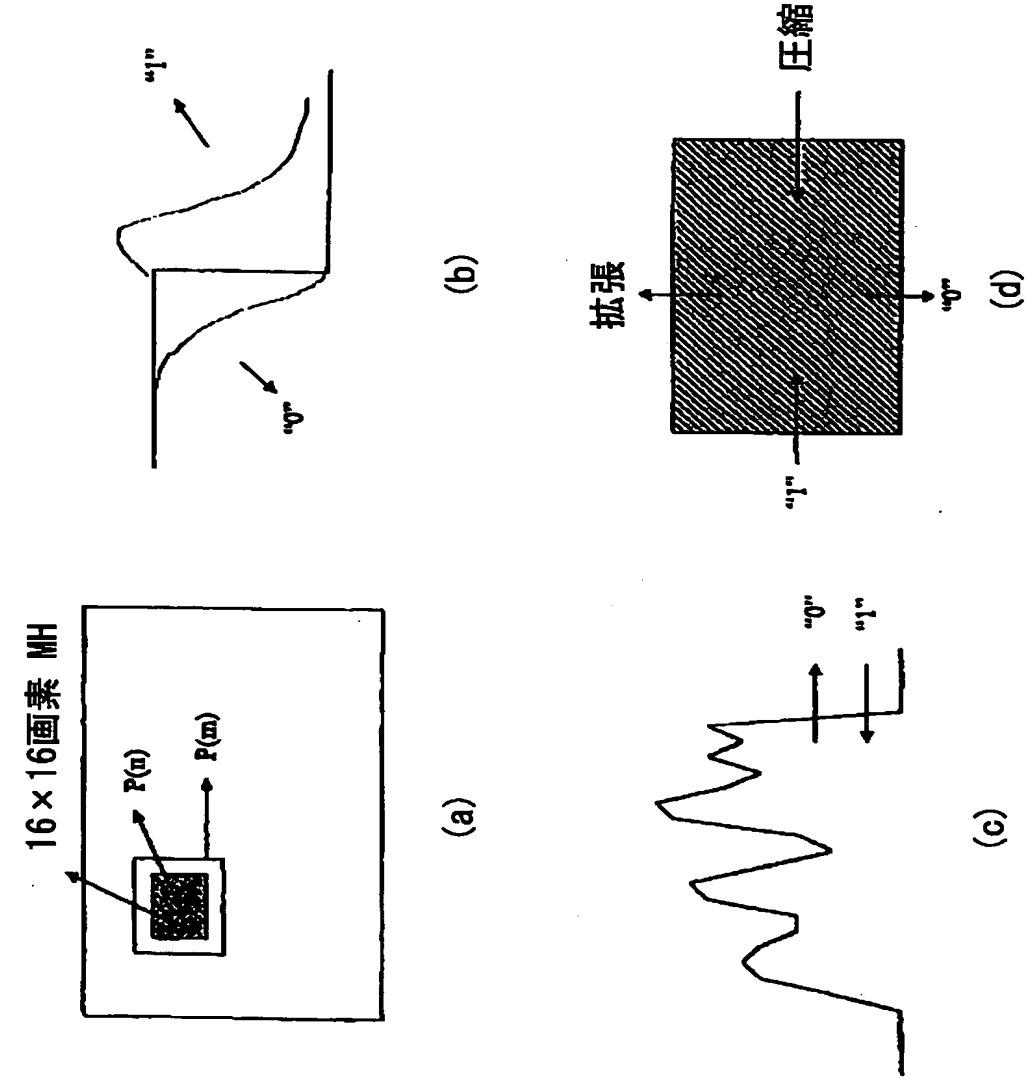
CDMA符号化処理：Wrは無作為に抽出された波形

【図９】

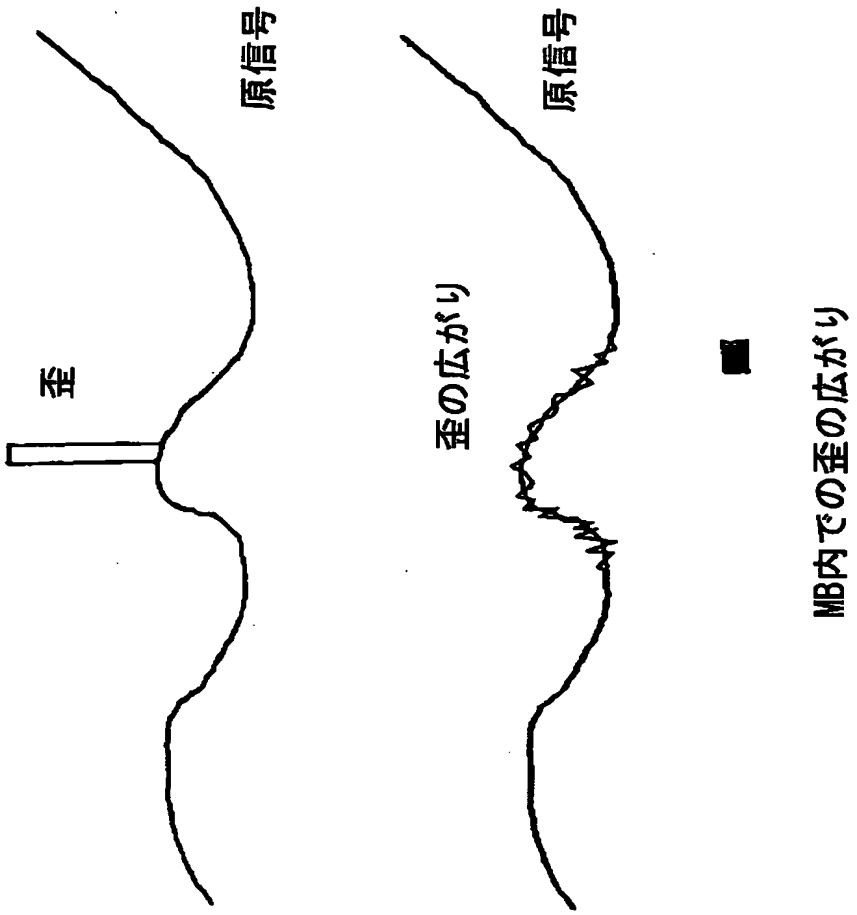


多重可変長並び換え：(a)入力ビット処理、(b)8種類の長さヘッドミュテータ
(c)P0とその逆数の例、(d)P0を用いたビット最終ブロックの処理、(e)ビット最終ブロック復元

【図１０】



【図１１】



【図１２】

```
for all the blocks of a MH
    Assign all the DCT coefficients to the Mhblock[0][block];

Initialize the random number generator with an input seed number;

for all the blocks of a MH {
  Select nDCT and sigma for luminance and chrominance blocks;
  Generate Arr[ ] of nDCT elements randomly with values of 0, 1 and 2
      according to the input ratio;

  for(i = 0; i < 64; i++){
    if( 1 <= i <= nDCT){
      if( Arr[i] == 1 ) {
         MHBlock[1][block][i] = MHBlock[0][block][i] + sigma;
         MHBlock[2][block][i] = MHBlock[0][block][i] - sigma;
      }
      else if ( Arr[i] == 2 )  {
         MHBlock[1][block][i] = MHBlock[0][block][i] - sigma;

         MHBlock[2][block][i] = MHBlock[0][block][i] + sigma;
      }
      else {
         MHBlock[1][block][i] = MHBlock[0][block][i];
         MHBlock[2][block][i] = MHBlock[0][block][i];
      }
    }
    else {
      MHBlock[1][block][i] = MHBlock[0][block][i];
      MHBlock[2][block][i] = MHBlock[0][block][i];
    }
  }
  Bound MHblock[1][block]and Mhblock[2][block] into legal DCT range;
}
```

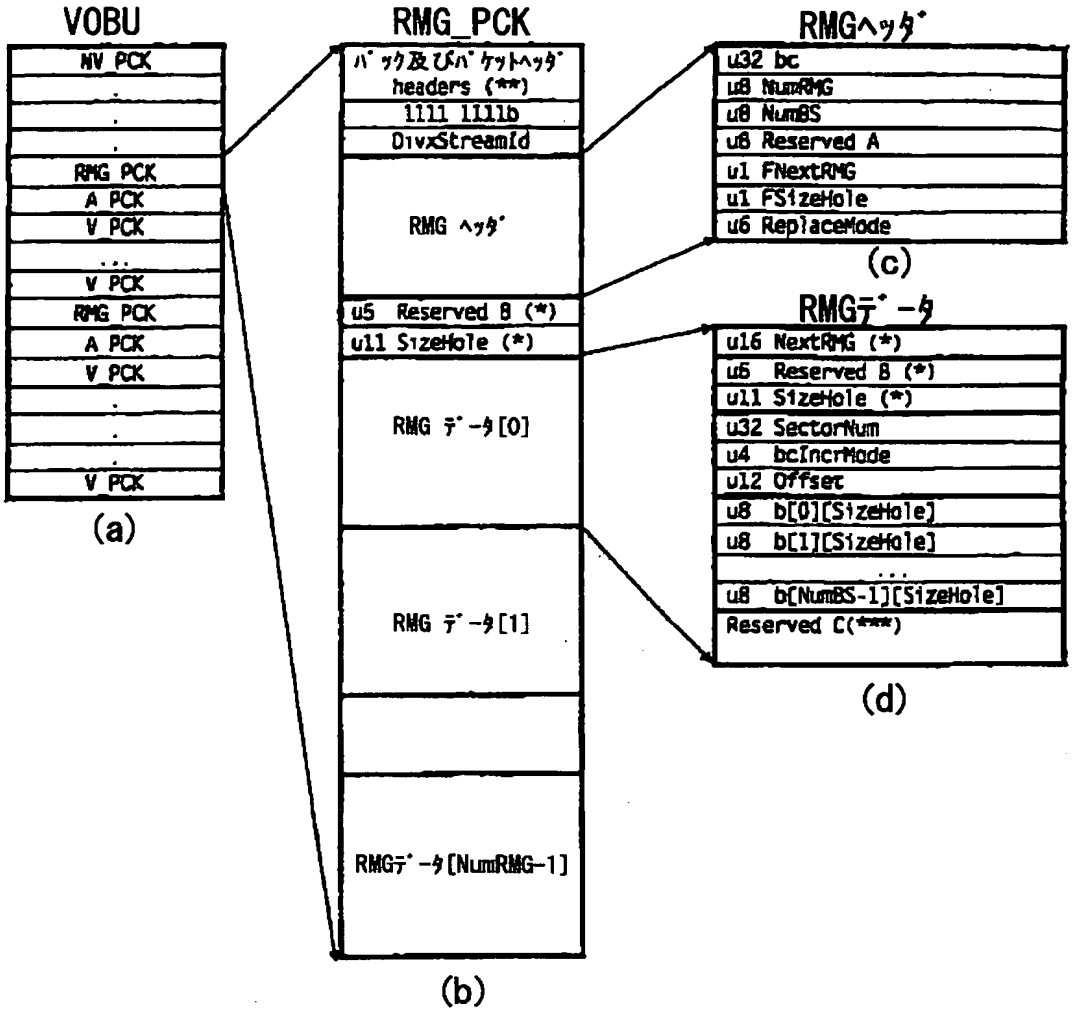
RMキャリアを生成するDCT法

【図１３】

0	4	[illegible]	-1	-7	-6	2	[illegible]	9	0	[illegible]	5	7	-4	3	0
0	-3	-2	5	0	0	4	[illegible]	9	0	-1	-2	0	-1	2	1
1	-2	[illegible]	1	3	0	3	2	1	-5	1	-2	-3	-3	-3	0
2	3	[illegible]	-2	[illegible]	-3	2	[illegible]	1	-5	1	[illegible]	1	-3	-4	3
1	1	7	1	-2	-4	2	2	9	1	-4	-1	3	5	2	4
1	[illegible]	[illegible]	-3	1	0	[illegible]	-2	2	1	-4	-4	-1	5	0	-1
4	[illegible]	0	-9	5	2	-2	[illegible]	-6	-4	-2	-5	-4	3	-5	-1
4	-6	-3	-8	3	1	[illegible]	0	1	-4	-2	-4	-4	6	-5	6
-2	3	6	-5	-3	0	3	-2	2	-2	-5	3	2	2	-3	-4
6	[illegible]	0	-7	[illegible]	-7	0	-2	2	0	-3	4	8	3	-2	0
1	-4	[illegible]	0	8	-5	1	0	-3	3	4	2	1	-2	5	4
-6	0	0	7	9	-1	5	2	-7	0	4	0	-5	-8	4	0
-2	5	1	6	3	0	4	0	-4	-2	-3	-1	3	-7	-5	-3
-2	-2	-1	1	5	-2	2	-4	2	0	0	-1	-2	-4	-3	1
-5	-4	-2	1	4	0	-1	-5	1	2	7	-1	-9	0	3	5
0	3	2	1	-4	0	-3	-3	-3	1	6	1	-3	6	-2	4

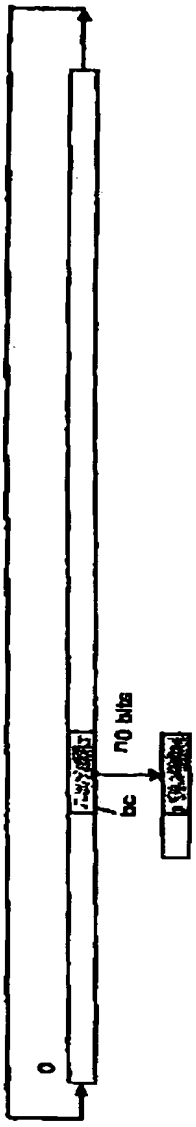
nDCT=40、sigma=5及び0, 1, 2の割合と2:1:1とするﾊﾟﾗﾒｰﾀにより
DCT法を用いた輝度ﾌﾞﾛｯｸ用画素領域の代表的ﾉｲｽﾞﾊﾟﾀｰﾝ

【図１４】



走行ﾏｰｸﾃﾞｰﾀｽﾄﾘｰﾑのﾃﾞｰﾀ配置及びﾃﾞｰﾀ構造：(a) VOBU内RMG_PCKのﾃﾞｰﾀ配置、(b) RMG_PCKのﾃﾞｰﾀ配置、(c) RMGﾍｯﾀﾞ、(d) RMGﾃﾞｰﾀ[0]
★印の変数は存在しない場合有り
★★：ﾊﾟｯｸ及びﾊﾟｹｯﾄﾍｯﾀﾞはV15-9のｾｸｼｮﾝ5, 2でDVD仕様に従う
★★★：この予約済領域はFNextRMG=1でない限り存在しない。FNextRMG=1の場合、NextRMGは次のRMGﾃﾞｰﾀの配置用に使用される。

【図１５】



Bits[ ]から送信用ビットの抽出

【図１６Ａ】

```
0   /* For: An algorithm of run-time module for a running mark system. */
1   /* By:  Siu-Leong Iu & Guillaume Mercier of Circuit City Stores Inc.. 4/28/98 */
2
3   int RunTimeModule(){
4     int i;
5     int errorNum = NoError;
6     u8 StreamId.SubStreamId.DivxStreamId;
7
8     while(readSector() != EndOfDVD){
9       errorNum = RunningMarking();
10      ErrorAction(errorNum);
11      readFromSector StreamId;
12      switch(StreamId){
13      case VideoStream:
14        Decode MPEG video bitstream and send result to video port;
15        break;
16      case AudioStream:
17        Decode MPEG audio bitstream and send result to audio port;
18        break;
19      case PrivateStream1:
20        readFromSector SubStreamId;
21        switch(SubStreamId){
22        case ProviderStream:
23           readFromSector DivxStreamId;
24           switch(DivxStreamId){
25           case RMGStream:
26             readFromSector bc, NumRMG, NumBS, FNextRMG, FSizeHole, ReplaceMode;
27             if(!FSizeHole)
28               readFromSector Reserved B, SizeHole;
29             for(i=0;i<NumRMG;i++)
30               ReadFromSector RMG_Data[i];
31               break;
32             case OtherDivxStreams:
33               break;
34             }
35             break;
36          case OtherSubStreams:
37             break;
38          }
39          break;
40       case PrivateStream2:
41          Process private stream 2;
42          break;
43       }
44     }
45     return(errorNum);
46  }
47
```

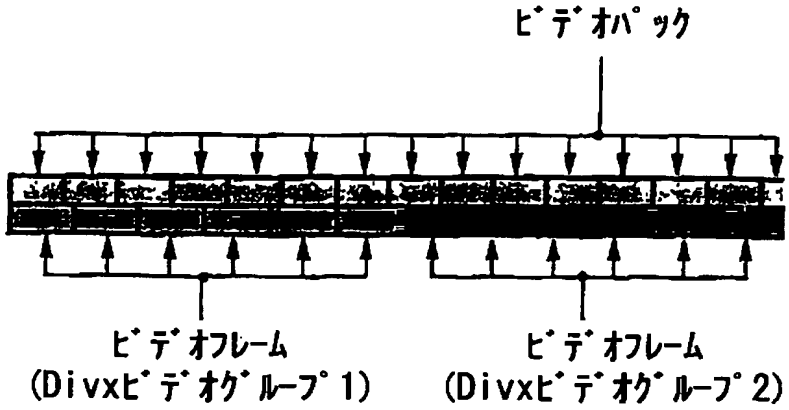
走行マークシステム用RMGランタイムモジュールのアルゴリズム

【図１６Ｂ】

```
48    int RunningMarking(){
49      int i,j;
50      int errorNum = NoError;
51
52      for(i=0; i<NumRMG; i++){
53        if(FNextRMG) NextRMG = RMG_Data[i].NextRMG;
54        else         calculate NextRMG;
55        if(FSizeHole)
56          SizeHole = RMG_Data[i].SizeHole;
57        SectorNum = RMG_Data[i].SectorNum;
58        if(SectorNum > CurrentSectorNum)
59          break;
60        bcIncrMode = RMG_Data[i].bcIncrMode;
61
62        if(SectorNum == CurrentSectorNum){
63          Offset = RMG_Data[i].Offset;
64          for(j=0; j<NumBS; j++)
65            b[j] = RMG_Data[i].b[j];
66          ReplaceMessageHoleData(ReplaceMode, bc);
67        }
68        update bc by bcIncrValue using bcIncrMode;
69      }
70
71      return(errorNum);
72    }
73
74    void ReplaceMessageHoleData(u6 Mode, u32 bc){
75      u16 sendBit =0;
76      locate MH bits of Message Hole using SectorNum and Offset;
77      switch(Mode){
78      case 0: /* Replace MH bits with b[0] */
79        replace MH bits with b[0].
80        break;
81      case 1: /* Replace MH bits with b[sendbit-1] */
82        extract n0 bits from Bits[ ] pointed by bc and store them to sendbit as in
83          figure 4;
84        if(0< sendbit && sendbit <= NumBS)
85          replace MH bits with b[sendbit-1];
86        break;
87      case 2: /* Replace MH bits with b[reverse(sendbit)-1] */
88        extract n0 bits from Bits[ ] pointed by bc and store them to sendbit as in
89          figure 4;
90        reverse each one of n0 least significant bits of sendbit;
91        if(0< sendbit && sendbit <= NumBS)
92          replace MH bits with b[sendbit-1];
93        break;
94      case OtherVersions;
95        break;
96      }
97    }
```

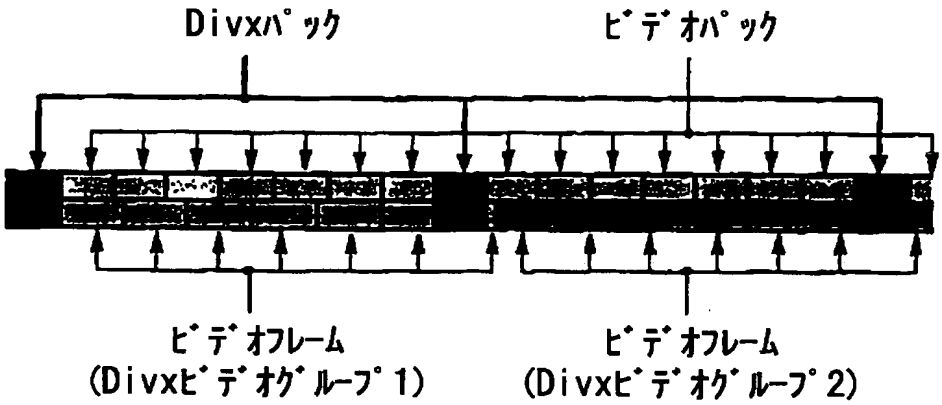
走行マークシステム用RMGランタイムモジュールのアルゴリズム

【図１７】



Divxパックなしビデオグループ配置

【図１８】



Divxパック挿入後のビデオグループ配置

【図１９】

```
1  Obtain values for the Width, Height, nFrames, numMHrow, and seed parameters;
2
3  MB_Width = Width / 16;
4  Make sure the numMHrow parameter is in a range of 0 to MB_Width;
5  Initialize the random number generator with the value of the seed parameter;
6
7  for (each frame in a total of nFrames frames) {
8       Clear (empty) the list of candidate MH addresses, MHAlist;
9       for (each MB row of the current frame) {
10              lowMBA = the MBA of the first MB in that row;
11              highMBA = the MBA of the last MB in that row;
12              while (less than numMHrow MHs have been selected on this MB row) {
13                      do {
14                              randMBA = a pseudo-random integer between lowMBA
                                        and highMBA;
15                      } while (randMBA is found in MHAlist);
16                      Append randMBA to MHAlist;
17              }
18      } /* next MB row */
19      Sort the MBA in MHAlist into ascending order;
20      Output the entire MHAlist;
21 } /* next frame */
```

gMHrowのアルゴリズム

【図２０】

但し
MHavlRow[i]:行iに使用可能なMH数
MHallocRow[I]:行Iに割り合てるMH数
MBtried:検索されたMB数
numMHreq:フレーム当たりの要求されたMH数
maxMHallow:1行内の最大可能MH数

```
1  initialize();
2
3  while (          the no. of MHs requested have not been obtained and
4                           the number of MBs examined is less than the maximum limit)
5  {
6        thisMBA = a randomly selected MBA of a MB that is available;
7        thisMB = the MB located at thisMBA;
8        MBrow = the MB row in which thisMB lies;
9        MBcol = the MB column in which thisMB lies;
10
11       if (most pixel values of thisMB are not within an acceptable range) {
12                Mark thisMB as unavailable;
13       } else    {

14                class_type = Classify_MacroBlock(thisMB);
15                if (class_type is TEXTURED) {
16                         if (MBrow doesn't already have the maximum no. of MHs per row on it)
   {
17                                  Mark thisMB as picked;
18                                  Mark the MHspace MBs on both sides of thisMB as
                                    unavailable;
19                         }
20                } else if (class_type is SMOOTH) {
21                         Mark thisMB as unavailable;
22                }
23       }
24 }
```

MB分類用擬似ｺｰﾄﾞ

【図２１】

```
initialize()

1     for (all MBs in the image) {
2       Mark them as available;
3     }
4
5     for (all MBs selected in the previous frame) {
6       Mark them as unavailable;
7     }
8
9     for (j = all the rows in the image) {
10      if (this row is the first or last row) {
11              for (all the MBs in the row)   {
12                      Mark them as unavailable;
13              }
14      } else  {
15              for (all MBs within BorderLimit distance from the border) {
16                      Mark them as unavailable;
17              }
18      }
19    }
```

分類ルーティーンにより呼び出されるinitialize( )用擬似コード

【図２２】

```
Classify MacroBlock():

Notations used:
        Ev - Vertical Energy
        Eh - horizontal energy
        Ed - diagonal energy
        Ea - Average energy

1       for (each of the 4 luminance 8×8 blocks in the MB) {
2         dct = the matrix of DCT coefficients of the block;

3         Eh        = (dct[0][2]² + dct[0][3]² + dct[0][4]²
4                   +  dct[1][2]² + dct[1][3]² + dct[1][4]²)/6;
5         Ev        = (dct [2][0]² + dct3][0]² + dct[4][0]²
6                   +  dct[2][1]² + dct[3][1]² + dct[4][1]²)/6;
7         Ed        = (dct[0][1]² + dct[1][0]² + dct[1][1]²
8                   +  dct[2][2]² + dct[2][3]² + dct[3][2]² + dct[3][3]² )/7;
9         Ea = (Eh + Ev + Ed)/3;
10        if (Ea > T1)
11                  Classify the 8×8 block as TEXTURED;
12      } /* next block */
13
14      if (3 or 4 blocks in the MB are classified to be TEXTURED)
15        The MB type is TEXTURED;
16      else
17        The MB type is SMOOTH;
```

MB分類用擬似コード

【図２３】

代表的セクションの図

【図２４】

```
rdbMHfn0 :

        Notations used:

        TotalMHsize - sum of the size of the MHs that have been selected;
        MaxMHsize - the maximum total size of all MHs allowed in a section;
        MHsize - the size of the selected MH;

1   initialize();
2
3   while (         there are MHs in the section that have not been selected
4                       and TotalMHsize < MaxMHsize)
5   {
6       for (thisFrame = all B frames in the section) {
7               Process the first frame every 1/startIncr times through this loop;
8               Process the last frame every 1/endIncr times through this loop;
9               if (there are MHs in thisFrame that have not been selected) {
10                      selectedMH = the smallest MH of the unselected ones in thisFrame;
11                      MHsize = the size of selectedMH;


12                      if (TotalMHsize + MHsize <= MaxMHsize) {
13                              Add MHsize to TotalMHsize;
14                              Select selectedMH as a final MH;
15                      } else {
16                              Break out of (terminate) the while loop;
17                      }
18              }
19      }
20  }
21
22  for (all MHs selected during the last iteration of the preceding while loop) {
23      deselect the MH;
24  }
25
26  step = round(no. of B frames per section / no. of MHs deselected);
27
28  for (thisFrame = every 'step' B frames in the section) {
29      if (there are MHs in thisFrame that have not been selected) {
30              selectedMH = the smallest MH of the unselected ones in thisFrame;
31              MHsize = the size of selectedMH;
32              if (TotalMHsize + MHsize <= MaxMHsize) {
33                      Add MHsize to TotalMHsize;
34                      Select selectedMH as a final MH;
35              }
36      } else  {
37              while (all B frames in the section have not been searched) {
38                      Search on both sides of thisFrame;
39                      if (a frame is found with an unselected MH) {
40                              selectedMH = the smallest MH of the unselected ones in
    thisFrame;
41                              MHsize = the size of selectedMH;
42                              if (TotalMHsize + MHsize <= MaxMHsize) {
43                                      Add MHsize to TotalMHsize;
44                                      Select selectedMH as a final MH;
45                              } else {
46                                      Break out of (terminate) the while loop;
47                              }
48                      }
49              }
50      }
51  }
52
53  Output all the MHs picked by this algorithm.
```

再配分用擬似ｺｰﾄﾞ

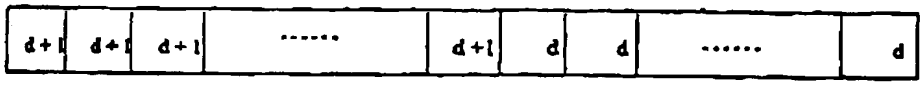
【図２５】

```
Initialize()

    Notations used:

1   if (  the     last     B     frame     of     the     previous     section     is
          the first B frame of the current section)
2   {

3       Add 1 to the no. of B frames to be processed;
4   }
5
6   for (all B frames in the section) {
7       for (all MHs in the B frame)  {
8           if (the MH is on a DVD boundary) {
9               Mark this MH as unavailable;
10          }
11      }
12  }
```

再配分関数により呼び出されるinitialize( )用擬似ｺｰﾄﾞ

【図２６】

現在行のMB: 0 1 2 ..... j ..... j+1 j+2



MH符号の余剰ﾋﾞｯﾄを補償する現在行MB用Mquant値の増加

【図２７】

```
1  for ( each row in the current frame ){
2    if ( this row is a MH row){
3      iterStatus = Inc_Mquant_Row;
4      for (iterCnt  =  0; iterCnt  < =  IterMax; iterCnt ++){
5        if ( ( iterCnt ! = 0 ) and (iterStatus ! = Final_Iteration) ){
6          increase the mquant value of  selected MB by 1;
7        }
8        Re-encode the whole row using the current mquant values for each MB;
9        if ( iterStatus == Final_Iteration )
10         Store the encoded bit stream for the whole row into the Row Buffer;
11       else
12         Only count the number of bits needed to re-encode row;
13       Byte Align the number of bits needed to re-encode row;
14       if (iterStatus == Final_Iteration){
15          Write video and mpeg information for MHs in the row to output Fifo;
16         Exit the iteration loop;
17         }
18       else if ( BitsDiff < = 0 )
19          iterStatus = Final_Iteration;
20        else
21          Select the next MB;
22     }
23     Perform bit stuffing to make OrigNumBits = RowBitCnt;
24     Replace original row with the encoded bits from Row Buffer;
25   }
26 }
```

第1のMPEG再符号化方法のｱﾙｺﾞﾘｽﾞﾑ

【図２８】

```
for ( each row in the current frame ){
  if ( this row is a MH row){
   iterStatus  = Inc_Mquant_Row;
   for (iterCnt  =  0; iterCnt  < =  IterMax; iterCnt ++){
    if ( iterCnt ! = 0){
      if (iterStatus == Inc_Mquant_Row)
        Increase the mquant values of all MBs in the row by delta;
      else if (iterStatus == Inc_Mquant_MB)
        Increase the mquant value of  selected MB by 1;
    }
    Re-encode the whole row using the current mquant values for each MB;
    if ( iterStatus == Final_Iteration )
      Store the encoded bit stream for the whole row into the Row Buffer;
    else
      Only count the number of bits needed to re-encode row;
    Byte Align the number of bits needed to re-encode row;
    if (iterStatus == Final_Iteration){
      Write video and mpeg information for MHs in row to output Fifo;
      Exit the iteration loop;
    }
    else if ( BitsDiff < 0 ){
      if (iterCnt ==0)
        iterStatus = Final_Iteration;
      else if (iterStatus == Inc_Mquant_Row){
        Decrease the mquant values of all MBs by delta;
        iterStatus = Inc_Mquant_MB;
        Select the first MB in the row;
      }
      else if (iterStatus == Inc_Mquant_MB)
        iterStatus == Final_Iteration;
    }
    else if ( BitsDiff == 0 )
        iterStatus = Final_Iteration;
    else if (iterStatus = Inc_Mquant_MB)
        Select the next MB;
   }

   Perform bit stuffing to make OrigNumBits = RowBitCnt;
   Replace original row with encoded bits from Row Buffer;
  }
}
```

第2のMPEG再符号化方法のｱﾙｺﾞﾘｽﾞﾑ

【図２９】

```
1  Sum = PrevBitCnt ;
2
3  for ( i = 0; i <  MB_width; i ++ ){
4     Sum = Sum + ( Bi(k) - Bi(k-1) );
5     if ( Sum < = RowBitCnt ){
6        j = i;
7        break;
8     }
9  }
```

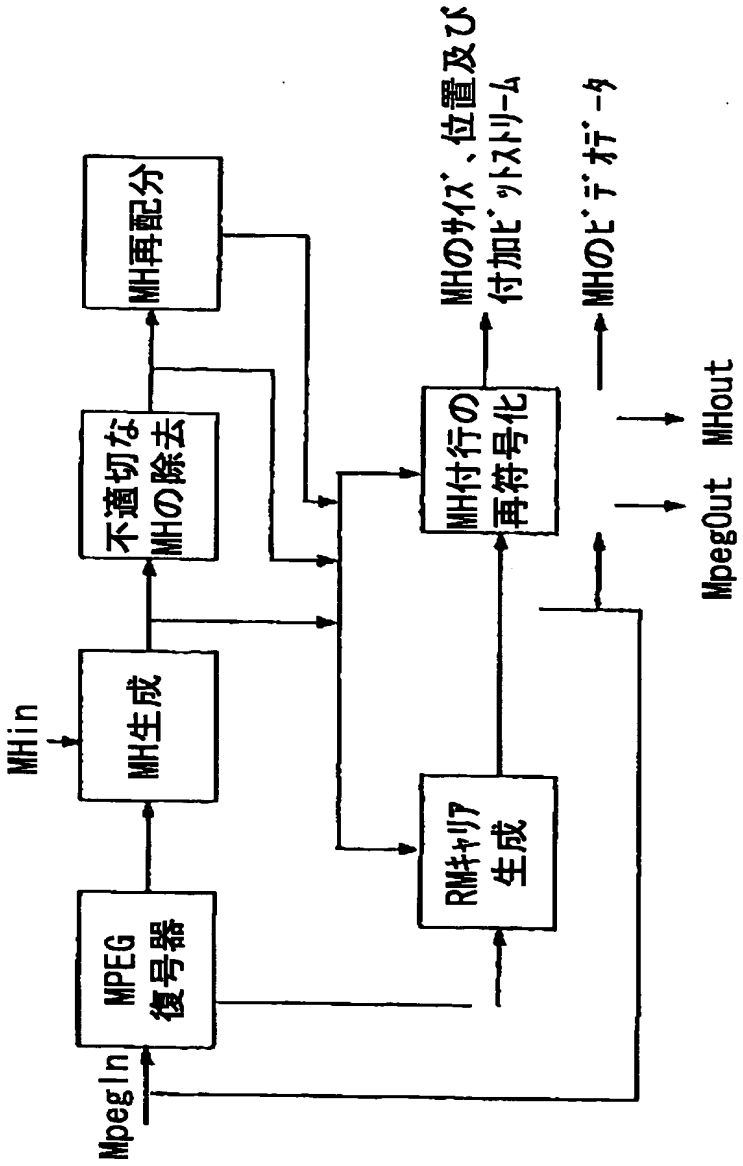
k=(d+2)時のk番目の反復の場合に
figure4.1内にjを配置するｱﾙｺﾞﾘｽﾞﾑ

【図３０】

```
1     for (  each row of MBs in the current frame ){
2       if ( this row is a MH row){
3         iterStatus  = Inc_Mquant_Row;
4         for (iterCnt  =  0; iterCnt  <=  IterMax; iterCnt ++){
5           if ( iterCnt != 0){
6              if (iterStatus == Inc_Mquant_Row)
7                Increase the mquant values of all MBs in the row by delta;
8              else if (iterStatus == Inc_Mquant_MB)
9                Increase the mquant value of selected MB by 1;
10          }
11          Re-encode the whole row using the current mquant values for each MB;
12          if ( iterStatus == Final_Iteration )
13            Store the encoded bit stream for the whole row into the Row Buffer;
14          else
15            Only count the number of bits needed to re-encode row;
16          Byte Align the number of bits needed to re-encode row;
17          if (iterStatus == Final_Iteration){
18            Write video and mpeg information for MHs in the row to output Fifo;
19            Exit the iteration loop;
20          }
21          else if ( BitsDiff < 0 ){
22              if (iterCnt ==0)
23                iterStatus = Final_Iteration;
24              else if (iterStatus == Inc_Mquant_Row){
25                iterStatus = Inc_Mquant_MB;
26                Find the value of j as in figure 4.1;
27                Decrease the mquant values of MBs by delta from j-th MB;
28                  Select j-th MB be the next MB to be updated;
29              }
30              else if (iterStatus == Inc_Mquant_MB)
31                iterStatus == Final_Iteration;
32            }
33            else if ( BitsDiff == 0)
34              iterStatus = Final_Iteration;
35            else if (iterStatus = Inc_Mquant_MB)
36              select next MB;
37            Save the value of the Row Buffer size to PrevBitCnt;
38          }
39          Perform bit stuffing to make OrigNumBits = RowBitCnt;
40          Replace original row with encoded bits from Row Buffer;
41        }
42      }
```

第3のMPEG再符号化方法のｱﾙｺﾞﾘｽﾞﾑ

【図３１】



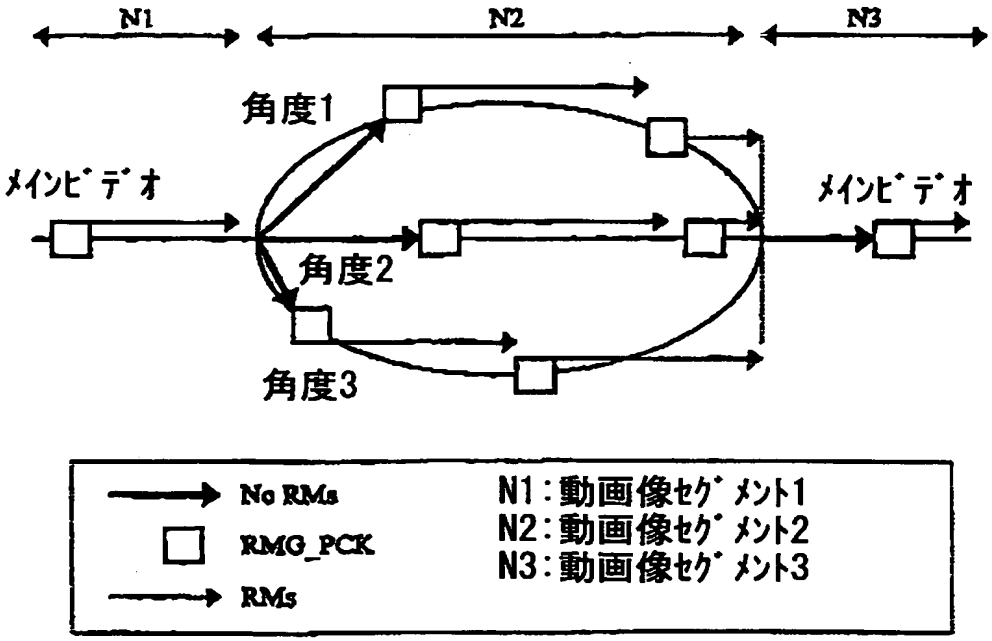
MH候補、RMキャリアの生成及び再符号化手順

【図３２】

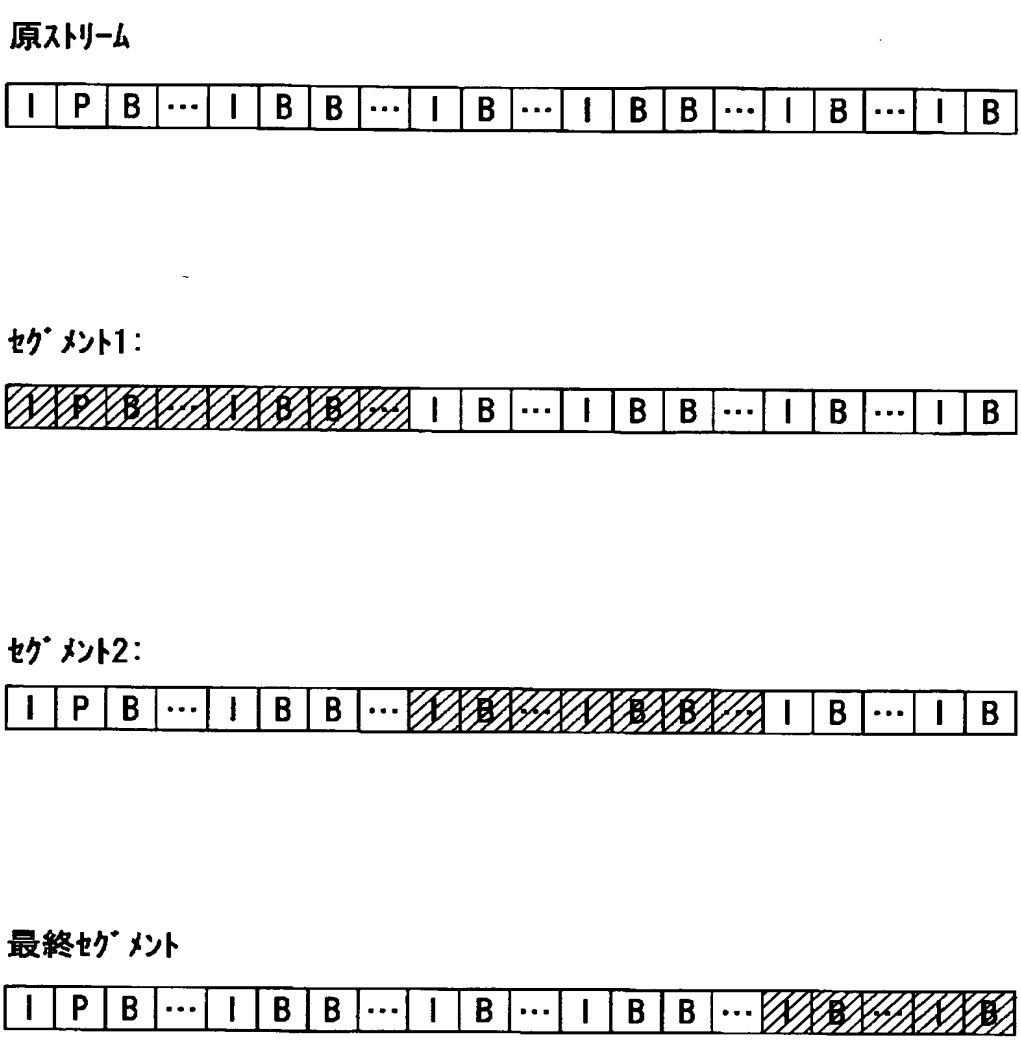
```
1       Read the 4 character re-encoding mode from the input parameter file;
2       for ( each frame from the image input file )
3       {
4          Decode the frame;
5          Get the encoding and display frame numbers;
6          if ( FirstTime)
7             allocate memory for MH information;
8          if ( end of sequence reached )
9          {
10            re-distribute the MHs in the remaining processed frames;
11            break;
12         }
13         if ( it is a B frame )
14         {
15            if ( MHProcMode[0] == GEN_MH_FROM_CLASSIFICATION )
16               generate MHs for frame with classification;
17            ReEncoding( );
18         }
19         if ( all frames in a Session have been processed )
20            re-distribute the MHs within all the processed frames in a
21            session;
22      }
```

GenRmﾌﾟﾛｸﾞﾗﾑのｱﾙｺﾞﾘｽﾞﾑ

【図３３】

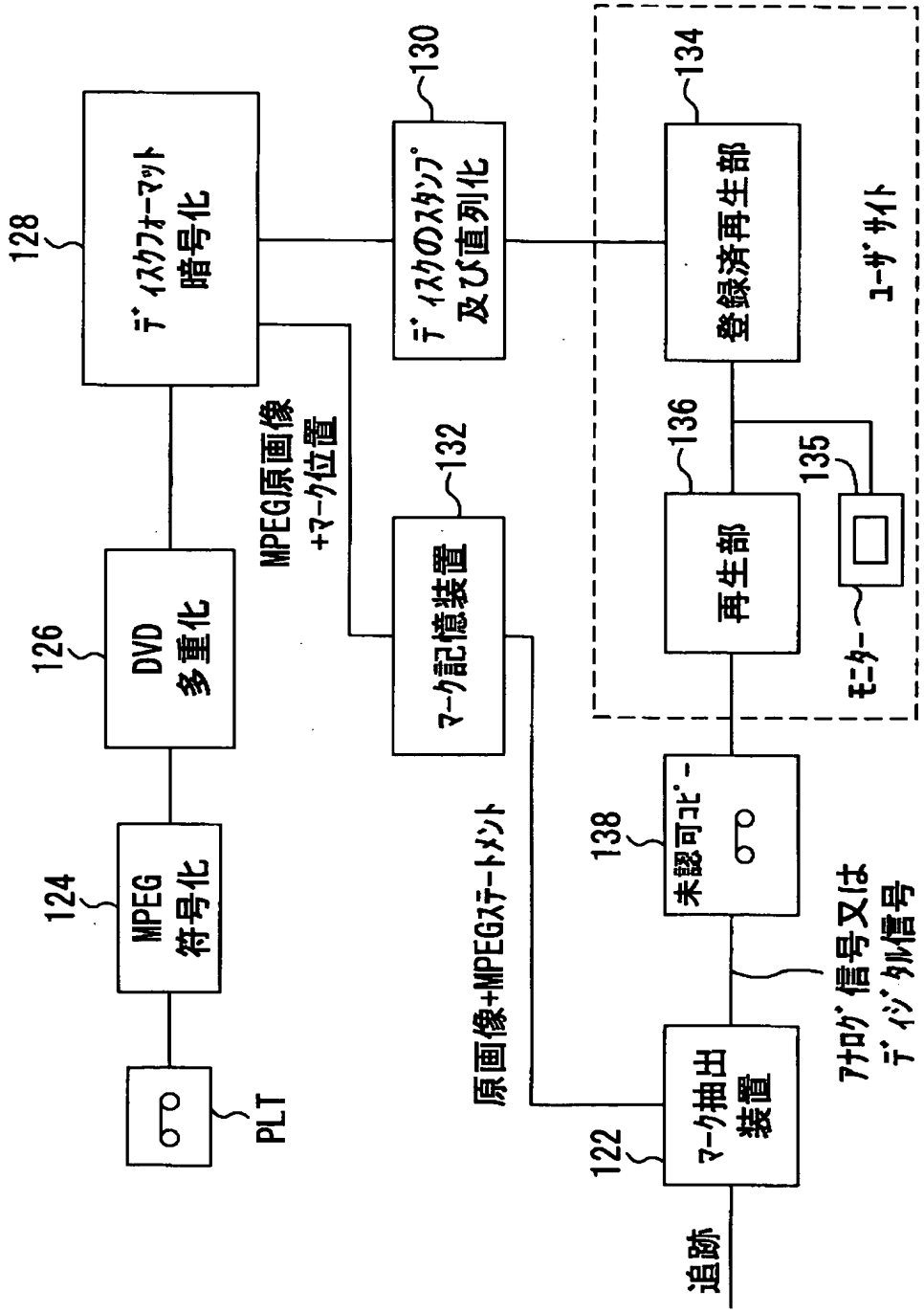


【図３４】


原ストリーム
I P B … I B B … I B … I B B … I B … I B
セグメント1：
I P B … I B B … I B … I B B … I B … I B
セグメント2：
I P B … I B B … I B … I B B … I B … I B
最終セグメント
I P B … I B B … I B … I B B … I B … I B

【手続補正書】特許協力条約第３４条補正の翻訳文提出書
【提出日】平成１２年８月７日（２０００．８．７）
【手続補正１】
【補正対象書類名】図面
【補正対象項目名】図１
【補正方法】変更
【補正内容】

【図１】


128
ディスクフォーマット
暗号化
130
ディスクのスタンプ
及び直列化
134
登録済再生部
1-ザサイト
MPEG原画像
+マーク位置
126
DVD
多重化
132
マーク記憶装置
136
再生部
135
モニター
124
MPEG
符号化
原画像+MPEGステートメント
138
未認可ユニット
PLT
122
マーク抽出
装置
アナログ信号又は
ディジタル信号
追跡

【国際調査報告】

INTERNATIONAL SEARCH REPORT

Inter·ional Application No
PCT/US 99/11797

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04N5/913

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 98 03014 A (PHILIPS ELECTRONICS N. V.) 22 January 1998 (1998-01-22) the whole document ---	1-3,5, 12,16, 17,19, 25,31, 36-38
A	WO 97 13248 A (PHILIPS ELECTRONICS N. V.) 10 April 1997 (1997-04-10) the whole document ---	1,3,4, 16,31, 36-39

-/—

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 September 1999	28/09/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Verleye, J

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Intern··nal Application No
PCT/US 99/11797

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 96 06503 A (MACROVISION CORPORATION) 29 February 1996 (1996-02-29) page 3, line 10 -page 7, line 12 page 8, line 6 -page 9, line 8; figures 1-3	1-3, 16-18, 31,33, 34,36,37
A	EP 0 107 567 A (THOMSON-CSF) 2 May 1984 (1984-05-02) page 3, line 14 -page 4, line 21; figure 3	1-3,16, 18,31, 36,37

1

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

formation on patent family members

Interr nal Application No
PCT/US 99/11797

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9803014 A	22-01-1998	EP 0940037 A	08-09-1999
		US 5933798 A	03-08-1999
WO 9713248 A	10-04-1997	CN 1166224 A	26-11-1997
		EP 0795174 A	17-09-1997
		JP 10510660 T	13-10-1998
WO 9606503 A	29-02-1996	US 5739864 A	14-04-1998
		AU 698870 B	12-11-1998
		AU 3370395 A	14-03-1996
		BR 9508624 A	18-11-1997
		CA 2195942 A	29-02-1996
		CN 1160467 A	24-09-1997
		EP 0777946 A	11-06-1997
		EP 0853433 A	15-07-1998
		JP 10504944 T	12-05-1998
		NZ 292020 A	28-10-1998
		US 5659613 A	19-08-1997
		US 5668603 A	16-09-1997
EP 107567 A	02-05-1984	FR 2534433 A	13-04-1984

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW

(72)発明者 コバド バグワディア
アメリカ合衆国 バージニア州 20165 スターリング、101、トリルムスクウエア 46229
(72)発明者 ギョーム メルシエ
アメリカ合衆国 バージニア州 22102 マックリーン、リンカンウェイ 1504、アパートメント 437
(72)発明者 シバ ラマドス
アメリカ合衆国 バージニア州 22102 マックリーン、リンカンウェイ 1504、アパートメント 437
(72)発明者 マイケル ベルジュロン
アメリカ合衆国 バージニア州 23192 モントピーリア、ウィスパリングスプリングレーン 15150
(72)発明者 ジャック エールハルト
アメリカ合衆国 バージニア州 23229－4935 リッチモンド、センチュリードライブ 7520

Fターム(参考) 5C053 FA13 GA07 GA11 GB06 GB15 GB37 JA24 KA21 KA24
5C059 KK43 MA22 MA23 MB27 RC04 RC09 RC31 RC35 RF02 RF05 RF18 RF27 RF30 SS13
5D044 AB05 AB07 BC06 CC04 DE03 DE40 GK08 GK17 HL08